

MAGAZINE

BSD

FOR NOVICE AND ADVANCED USERS

FREEBSD BASED DUAL-CONTROLLER
STORAGE SYSTEM CONCEPT

BENCHMARKS MacOSx VS FREEBSD

GHOSTBSD

EASY TO USE YET POWERFUL

SERVER AUTOMATION

FOR NODEJS OVER SSH WITH NODEMIRAL

FREENAS GETTING STARTED GUIDE
PART 1, PLANNING AND INSTALLATION

INTERVIEW WITH
ERIC TURGEON

VOL 10, NO. 05
ISSUE 05/2016 (81)
1898-9144

SOMETHING XL IS COMING IN APRIL



ENTERPRISE-CLASS HARDWARE, RUNNING
THE WORLD'S MOST POPULAR OPEN SOURCE
STORAGE OPERATING SYSTEM.

For more information on the FreeNAS Mini,
visit **ixsystems.com/mini** today.

Then, check back in April for something XL...

Dear Readers,

The beautiful month of May is behind us and we are heading into the holiday period. Where are you going to spend your vacation?

What's more, in the beginning of July, our American readers will celebrate Independence Day. We would like to wish you all the best for you and your families! We hope you will spend this day in peace, happiness, and joy.

Moving on to this issue, it opens with industry news, as always. Next, you will read "FreeBSD Based Dual-Controller Storage System Concept" by Mikhail E. Zakharov. This is the second great article out of three by this author.

We are proud to introduce you to Natalia Portillo. She made a great comparison of MacOS and FreeBSD in her article "Benchmarks MacOS X vs FreeBSD".

After her article, we have another (after the previous issue), a GhostBSD related article "GhostBSD - Easy to Use, Yet Powerful" by Kalin Staykov. If you like this BSD, you will enjoy our interview with Eric Turgeon as well.

"Server Automation for NodeJS over SSH with Nodemiral" by Ray Mahangoe explains what Nodemiral is and how to install it.

iXsystems prepared a series of articles for you as well. Mark VonFange has shared "FreeNAS Getting Started Guide: Part 1, Planning and Installation" with us as the first installment in a 3 part series.

Finally, as always, you can find a great article by Rob Somerville about Barclays bank.

Enjoy the reading and have a beautiful June! "Don't count the days, make the days count" - Muhammad Ali.

Marta & BSD Team

MAGAZINE **BSD**

Editor in Chief:

Marta Ziemianowicz

marta.ziemianowicz@software.com.pl

Contributing:

Mikhail E. Zakharov, NataliaPortillo, Ray Mahangoe, Kalin Staykov, Mark VonFange, Eric Tugeon, and Rob Somerville.

Top Betatesters & Proofreaders:

Annie Zhang, Denise Ebery, Eric Geissinger, Luca Ferrari, Imad Soltani, Olaoluwa Omokanwaye, Radjis Mahangoe, Mani Kanth and Mark VonFange.

Special Thanks:

Annie Zhang

Denise Ebery

DTP:

Marta Ziemianowicz

Senior Consultant/Publisher:

Paweł Marciniak

pawel@software.com.pl

CEO:

Joanna Kretowicz

joanna.kretowicz@software.com.pl

Publisher:

Hakin9 Media SK 02-676 Warsaw, Poland Postepu 17D Poland worldwide publishing editors@bsdmag.org www.bsdmag.org

Hakin9 Media SK is looking for partners from all over the world. If you are interested in cooperation with us, please contact us via e-mail: editors@bsdmag.org.

All trademarks presented in the magazine were used only for informative purposes. All rights to trademarks presented in the magazine are reserved by the companies which own them.

CONTENTS

News

[BSD World Monthly News](#) **4**

by Marta Ziemianowicz

This column presents the latest news coverage of breaking news events, product releases and trending topics.

FreeBSD Corner

[FreeBSD Based Dual-Controller Storage System Concept](#) **15**

by Mikhail E. Zakharov

Nowadays, most of the modern and powerful block-level storage systems around the world are built to work in expensive Fibre Channel or cheaper iSCSI SAN environments. Independent of their class, capacity and performance, they are created on well-known principles, technologies and architectures. Furthermore, some of these systems are using common servers for their controller hardware with Linux or even AIX as storage operating systems on-board.

Fight Club

[Benchmarks: MacOS X vs FreeBSD](#) **46**

by Natalia Portillo

Imagine you've got an old Macintosh sitting around and you think you may be able to use it for other purposes. You have some expertise on FreeBSD and feel quite confident using it. In this article, I'll try to explore what's better, to install Mac OS X until Apple stops supporting your exact model, or move on to FreeBSD. For this purpose, I'll explore several advantages and disadvantages for diverse use case scenarios as well as pure benchmarking for both.

NodeJS

[Server Automation for NodeJS over SSH with Nodemiral](#) **61**

by Ray Mahangoe

Nodemiral is like Ansible, an automation tool based on Javascript/NodeJS. While Ansible has been around for a while, Nodemiral is still young (though Javascript has been in the loop for a while). Why do I

use Nodemiral? Well, I've been using Ansible for a while and I like the way Ansible works, like access with ssh, playbooks and run script on a server, etc., and there is no need to install a client on every server.

GhostBSD

[GhostBSD - Easy to Use, Yet Powerful](#) **68**

by Kalin Staykov

When I first heard about GhostBSD, I thought it may involve a shady distribution that is all about security. Okay, in fact, at first I thought about actual ghosts, but let's not dive into that. The name comes from "Gnome hack operating system technology BSD". This project is all about putting a nice desktop environment with all the security perks of having a BSD system under the hood.

FreeNAS

[FreeNAS Getting Started Guide: Part 1, Planning and Installation](#) **77**

by Mark VonFange

This article is intended to serve as an introductory guide to assist FreeNAS users in planning, installation, configuration and administration for their FreeNAS storage systems. Each category will include a high level discussion covering the basics of what is needed, with applicable screenshots.

Interview

[Interview with Eric Turgeon, Founder and Leader of GhostBSD](#) **88**

by Marta Ziemianowicz, Marta Strzelec & Marta Si-enicka

[Rob's Column](#) **90**

by Rob Somerville

Barclays bank, as part of their Life Skills television and Internet campaign, are advising those entering the job market to use more professional email addresses. In light of their involvement in the Libor scandal, where they attempted to manipulate the benchmark inter-bank borrowing rate, can we take this advice seriously?

BSD Certification

The BSD Certification Group Inc. (BSDCG) is a non-profit organization committed to creating and maintaining a global certification standard for system administration on BSD based operating systems.

? WHAT CERTIFICATIONS ARE AVAILABLE?

BSDA: Entry-level certification suited for candidates with a general Unix background and at least six months of experience with BSD systems.

BDSP: Advanced certification for senior system administrators with at least three years of experience on BSD systems. Successful BDSP candidates are able to demonstrate strong to expert skills in BSD Unix system administration.

✓ WHERE CAN I GET CERTIFIED?

We're pleased to announce that after 7 months of negotiations and the work required to make the exam available in a computer based format, that the BSDA exam is now available at several hundred testing centers around the world. Paper based BSDA exams cost \$75 USD. Computer based BSDA exams cost \$150 USD. The price of the BDSP exams are yet to be determined.

Payments are made through our registration website:
<https://register.bsdcertification.org/register/payment>

i WHERE CAN I GET MORE INFORMATION?

More information and links to our mailing lists, LinkedIn groups, and Facebook group are available at our website:
<http://www.bsdcertification.org>

Registration for upcoming exam events is available at our registration website:
<https://register.bsdcertification.org/register/get-a-bsdcg-id>

Indiegogo DinoBSD for the mobile



User George Hadjichristofi is starting a campaign to bring FreeBSD to the mobile device.

Welcome to my campaign to complete the DinoBSD: An OS for smartphones. DinoBSD is a fresh, easy and secure OS based on FreeBSD.

The funds raised here will be used to build a DinoBSD OS

What We Need & What You Get

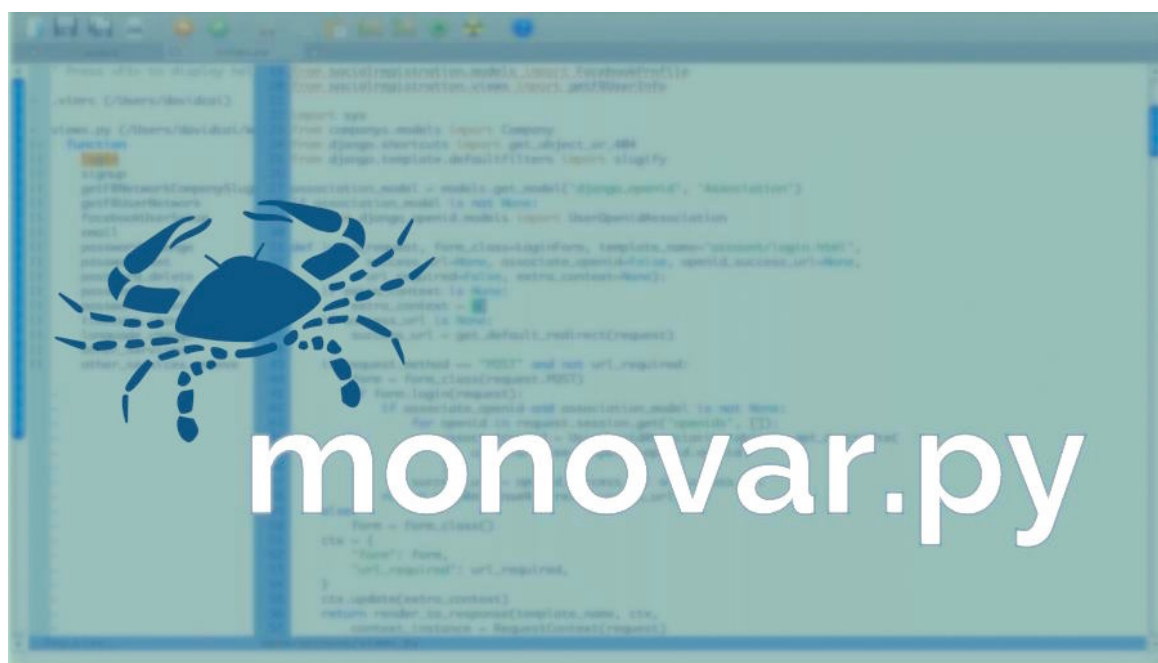
I will need to do more development to finish the first beta version and testing on smartphones. I would like to find a company to build the hardware to test DinoBSD which will probably be covered under intellectual property law (patents/ trademarks / copyrights) If I don't reach the entire goal, I will continue the project, but slower.

DinoBSD:

<https://www.indiegogo.com/projects/dinobsd-2#/>

<https://www.freebsdnews.com/2016/05/27/httpwww-indiegogo-comprojectsdinobsd-2/>

Wayland's Weston Nukes Its Raspberry Pi Backend/Renderer



Monovar is a sophisticated algorithm to detect single nucleotide variants (SVNs) in cancer cells. Written in Python, this program can help in providing a more personalized treatment to cancer patients by pinpointing important variations in a single cancer cell.

The current technology being used to detect DNA mutations in cancer cells analyzes millions of cells to arrive at some definite conclusions. To make this process more efficient, the re-

searchers at The University of Texas MD Anderson Cancer Center have come up with a new method.

Called Monovar Method, this computer program can spot DNA mutations taking place in a single cancer cell and pinpoint important variations.

This new and sophisticated program can help our doctors take a more personalized approach to cancer treatment before going for methods like chemotherapy.

The new Monovar method uses a technology called single cell sequencing (SCS). SCS also finds its applications in other fields of medicine, like microbiology, neurobiology, and immunology.

Monovar is able to detect single nucleotide variants (SNVs), which is a very minute kind of DNA variation. SNVs play an important role in cancer treatment as they affect how a patient develops the disease and responds to the medication.

By helping in detecting SNVs more accurately, more personalized treatments could act as a better choice.

You can take a look at the Monovar program online and see how it works. Written in Python, here's what the Bitbucket description of Monovar program says:

Monovar is a single nucleotide variant (SNV) detection and genotyping algorithm for single-cell DNA sequencing data. It takes a list of bam files as input and outputs a vcf file containing the detected SNVs.

<http://fossbytes.com/monovar-python-program-cancer-detection-svn/>

GhostBSD 10.3 ALPHA 2 available



The developers of GhostBSD have made available the second ALPHA for version 10.3.

This second ALPHA development release is for testing and debugging new features in GhostBSD 10.3, MATE and XFCE is available on SourceForge for the i386, amd64, and amd64-uefi architectures.

New feature.

- UEFI is now supported for 64bit only.

What has been fixed.

- Gbi log and partition data base is now deleting when closing gbi.
- Resetting partition data base when going back to the installation type.
- 4K Partition Alignment.
- Networkmgr full SSID name.
- System Update duplicating the whole install under /boot/kernel.old.
- Gbi “back” button error.
- sudo configuration.
- Wifi down by default.
- /etc/localtime not created by installer.
- Locales are not correctly set up on installation.
- missing `kern.vty=vt` setting in `grub.cf`.

Official announcement: <http://ghostbsd.org/10.3-ALPHA2>

<https://www.freebsdnews.com/2016/05/27/ghostbsd-10-3-alpha-2-available/>

Snaps on NAS: IoT apps for your private network

QNAP selects snaps and Ubuntu to bring IoT apps to its NAS



QNAP announced that they were adopting snaps as the application format of choice for their NAS going forward. Behind this decision are two factors, the ease of development of snaps and the universality of snaps, especially to create IoT applications.

NAS has been around for a while, and have been used across a wide array of use cases, from media servers in tech-savvy households to remote file server by SMBs. QNAP has been a pioneer in this space, offering an appstore for their NAS with hundred of apps: local web servers (WordPress, Drupal), Media server (Kodi), Cloud backup ...

By adopting snaps as the preferred application format for their NAS going forward, QNAP is looking to surf on the growing popularity of snaps across Ubuntu desktop and Ubuntu Core in IoT. They're keen to leverage universality of snaps, which can be deployed from Ubuntu desktops to Ubuntu Core IoT Gateways. But also the simplicity of creating snaps, with snapcraft a tool that makes it simple to build secure, contained applications from source easily.

<https://insights.ubuntu.com/2016/05/31/snaps-on-nas-iot-apps-for-your-private-network/>

Wayland's Weston Nukes Its Raspberry Pi Backend/Renderer



Upstream Wayland developers have decided to drop the specialized Raspberry Pi back-end and renderer from the Weston compositor code-base.

Back in 2012 was the premiere of this Raspberry Pi back-end for Weston that made use of the DispmanX API for initializing the display and other changes, compared to the more traditional DRM back-end for Weston. This Raspberry Pi code in Weston hasn't received much attention lately to Weston and has now been dropped.

Pekka noted, "The rpi-backend is a good example of how using an API that is only available for specific hardware, even more so as it is only available with a proprietary driver stack, is not maintainable in the long run. Most developers working on Weston either just cannot, or cannot bother to test things also on the RPi. Breakage creeps in without anyone noticing."

http://www.phoronix.com/scan.php?page=news_item&px=Weston-Nukes-RPi-Code

New Gentoo LiveDVD Released, Powered By Linux 4.5 & Supports ZFS



It's been quite a while since the last Gentoo LiveDVD release, but a new image has surfaced this weekend as Gentoo 20160514 and codenamed the "Choice Edition" release.

This first Gentoo release of 2016 is powered by the Linux 4.5 kernel and features X.Org Server 1.18.3, KDE Plasma 5.6.2, Firefox 45.0, LibreOffice 5.1, GIMP 2.9.2, and many other packaging updates.

Aside from updated packages, this Gentoo "Choice Edition" LiveDVD features ZFS On Linux support, UEFI support, and writable file-systems using AuFS.

This is the first official Gentoo LiveDVD release since 2014.

For those wanting to remember what Gentoo looked like ten years ago... or in 2008 and 2009.

https://www.phoronix.com/scan.php?page=news_item&px=Gentoo-May-2016-Release

Pyston 0.5 Released As A Faster Python JIT

The Dropbox engineers working on their Pyston project as a high-performance JIT implementation today announced version 0.5 of the software.

Pyston 0.5 now makes use of reference counting rather than tracing garbage collectors, support for running NumPy unmodified, signal handling support, frame introspection, and many other changes.

All of the details on Pyston 0.5 -- including their motives for abandoning the tracing GC and moving to refcounting -- is described via this blog post announcement on the Pyston project site.

http://www.phoronix.com/scan.php?page=news_item&px=Pyston-0.5-Python

pfSense 2.3.1-RELEASE now available



The developers of pfSense have made available version 2.3.1 release. Download the update or install file.

We are happy to announce the release of pfSense® software version 2.3.1!

This is a maintenance release in the 2.3.x series, bringing a number of bug fixes, two security fixes in the GUI, as well as security fixes for OpenSSL, OpenVPN and FreeBSD atkbd and sendmsg. The full list of changes is on the 2.3.1 New Features and Changes page.

This release includes a total of 103 bug fixes. 79 regressions in 2.3 have been fixed, mostly minor issues in the new GUI. Several of these are significant issues, and have resolved nearly all the post-upgrade problems encountered in 2.3-RELEASE. 24 issues affecting 2.2.x and prior versions have also been fixed.

If you haven't yet caught up on the changes in 2.3.x, check out the Features and Highlights video. Past blog posts have covered some of the changes, such as the performance improvements from tryforward, and the webGUI update.

Official announcement: <https://blog.pfsense.org/?p=2050>

<https://www.freebsdnews.com/2016/05/20/pfsense-2-3-1-release-now-available/>

Nginx 1.11 Web Server Released

Version 1.11 of the open-source, high-performance Nginx web-server is now available.

Nginx 1.11 presents a new transparent parameter for several options, support for loading multiple certificates of different types, various other security-related changes, a \$proxy_protocol_port variable, some HTTP/2 changes, and more.

The complete list of nginx 1.11 changes can be found via this file. Download the open-source Nginx 1.11 via Nginx.org.

http://www.phoronix.com/scan.php?page=news_item&px=Nginx-1.11-Released

Citrix dodges death, returns with bigger XenServer and NetScaler

Drinking the software-defined big data kool aid through one pane of glass

Citrix has unified its networking products and made big additions to its virtualisation stack.

Last things first: XenServer is now in version 7, which means it gains support for Intel's Iris Pro graphics technology. In theory, that makes XenServer a better platform for graphics-intensive desktop virtualisation (VDI) chores. Citrix thinks there's lots of you who would love the chance to banish workstation fleets and replace them with VDI, or just make sure VDI can keep up with the requirements of video-guzzling users.

Citrix and Microsoft have been close for ages and get closer with this release, which adds support for Server Message Block (SMB) for virtual machine storage. The upshot of this is Citrix users in big Windows shops can now point XenServer at shared storage managed by Windows Server, which should help users to make the most of storage assets.

Citrix is taking advantage of the combination of System Center Configuration Manager and Windows Server Update Services, as together they make it possible to update XenServer with Windows Update.

Everything's bigger this time around: host RAM goes up from 1TB to 5TB and 288 CPU cores are now permitted, double the previous count. VRAM now reaches 1.5TB, an eightfold increase.

There's also a new agentless API allowing inspection of guest VMs' memory, the better to make sure they're not running something nasty.

Cloud-native types may appreciate support for managing Docker from within Xen Server.

Nothing radical, therefore, but some nice bits for those running all sorts of workloads on Xen Server. Citrix's virtualisation strategy sees it prioritise its own products – XenDesktop and XenApp – in the hope of selling a stack. This release won't hurt that strategy and may give comfort to those hoping to continue using XenServer for other stuff, plus the beginning of a direction on containers.

Citrix's other strong suit is networking and the company's decided it's time to bring all its offerings into a single management console. The new “NetScaler Management and Analytics System” therefore gives users one tool with which to manage the company's CloudBridge software-defined WAN, NetScaler application delivery controller and VPN products.

CloudBridge was previously a discrete product but is now re-christened NetScaler SDWAN. The whole suite is now backed by better analytics and automation, plus templates that let

you quickly cook a secure software-defined network and network optimisation environment for different applications.

The mantra, as is the case across the industry, is that in these cloudy times we all need to be able to create networks that span our own bit barns and public clouds.

Citrix grew in its most recently-reported quarter, a result felt to represent a slightly-unexpected turnaround after years of confused strategy and red ink. The announcements above, the cute adoption of the Raspberry Pi as a thin client and a new workspace-as-service push surely mean the company has moved off the “not dead yet” pile and closer to many shopping lists. ®

http://www.theregister.co.uk/2016/05/25/citrix_dodges_death_returns_with_bigger_xenserver_and_netscaler/

Android might be on the way to the Raspberry Pi

A new tree sprouts on android.googlesource.com

Evidence for the development can be found here in the recently-created `android/device/pifoundation/rpi3` directory at android.googlesource.com.

It's an empty tree and has been since its creation on April 19th, 2016 by someone called Thomas Joseph Avila who has a `google.com` email address.

Google's open source Android project already has code for Intel's Edison, Arduino-powered accessories and even TI's Panda single-board computers. Taking a slice of Pi is therefore not an outlandish move.

There's no sign that work is in progress or of a timeline having been set for Android-on-Pi's completion.

If and when it's done, it will be an intriguing offer. Android, of course, has a colossal collection of apps and is rather more familiar to many people than Raspbian and perhaps more approachable than even the NOOBS OS installer.

The Register suspects Google and the Raspberry Pi foundation would not mind if Android on Pi gives more people a reason to acquire the machine. Or is Google instead trying to make Android a more viable target for the customised industrial Pis the Foundation is pitching as an Internet of Things thing? ®

http://www.theregister.co.uk/2016/05/26/android_might_be_on_the_way_to_the_raspberry_pi/

Microsoft has created its own FreeBSD image

Redmond will support it inside Azure and send code back to the FreeBSD Foundation



Microsoft has created its own cut of FreeBSD 10.3 in order to make the OS available and supported in Azure.

Jason Anderson, principal PM manager at Microsoft's Open Source Technology Center says Redmond "took on the work of building, testing, releasing and maintaining the image"

so it could "ensure our customers have an enterprise SLA for their FreeBSD VMs running in Azure".

Microsoft did so "to remove that burden" from the FreeBSD Foundation, which relies on community contributions.

Redmond is not keeping its work on FreeBSD to itself: Anderson says "the majority of the investments we make at the kernel level to enable network and storage performance were up-streamed into the FreeBSD 10.3 release, so anyone who downloads a FreeBSD 10.3 image from the FreeBSD Foundation will get those investments from Microsoft built in to the OS."

Code will flow both ways: Anderson says "... our intent is to stay current and make available the latest releases shortly after they are released by the FreeBSD Release Engineering team. We are continuing to make investments to further tune performance on storage, as well as adding new Hyper-V features – stay tuned for more information on this!"

Microsoft says it will support its distribution when run in Azure.

Redmond's rationale for the release is that plenty of software vendors use FreeBSD as the OS for software appliances. That reasoning was behind Microsoft's 2012 decision to ensure FreeBSD could run as a guest OS under Hyper-V. In your own bit barns, your guest OSes are your own problem. Microsoft clearly decided it needed something more predictable for Azure, although it has in the past allowed custom FreeBSDs to run as cloudy VMs.

Of course Microsoft has also allowed Linux on Azure VMs for years, so news of the FreeBSD effort feels like an effort to ensure the platforms cloud users want are available rather than a startling embrace of open source to rank with Azure's don't-call-it-a-Linux-for-switches or the announcement of SQL Server for Linux.

But it's still just a little surprising to see Microsoft wade into development of FreeBSD: this is not your father's Microsoft.

One last thing: when Microsoft announced it would ensure FreeBSD runs on Hyper-V, NetApp was one of its collaborators. NetApp knows FreeBSD inside out, because Data ONTAP is built on it. But NetApp is absent from the vendors listed in Microsoft's announcement of its FreeBSD efforts. Which might put the kybosh on our imagined cloud-spanning software-defined NetApp rigs.

®

http://www.theregister.co.uk/2016/06/09/microsoft_freebsd/

Great Specials

On FreeBSD® & PC-BSD® Merchandise

Give us a call & ask about our
SOFTWARE BUNDLES

1.925.240.6652

\$39.95

FreeBSD 9.1 Jewel Case CD Set
or FreeBSD 9.1 DVD

\$29.95

PC-BSD 9.1 DVD

\$49.95

The PC-BSD 9.0 Users Handbook
PC-BSD 9.1 DVD

\$99.95

The FreeBSD CD or DVD Bundle

Inside each CD/DVD Bundle, you'll find:
FreeBSD Handbook, 3rd Edition
Users Guide FreeBSD Handbook, 3rd Edition, Admin Guide
FreeBSD 9.1 CD or DVD set
FreeBSD Toolkit DVD



Stylish Dress Attire
Look Your Professional Best



Comfy Apparel
Stay Warm in Zip Ups & Pullovers

T-Shirts
Lots of Styles to Choose From

FreeBSD 9.1 Jewel Case CD/DVD \$39.95

CD Set Contains:

- Disc 1** Installation Boot LiveCD (i386)
- Disc 2** Essential Packages Xorg (i386)
- Disc 3** Essential Packages, GNOME2 (i386)
- Disc 4** Essential Packages (i386)

FreeBSD 9.0 CD \$39.95

FreeBSD 9.0 DVD \$39.95

FreeBSD Subscriptions

Save time and \$\$\$ by subscribing to regular updates of FreeBSD

FreeBSD Subscription, start with CD 9.1 \$29.95

FreeBSD Subscription, start with DVD 9.1 \$29.95

FreeBSD Subscription, start with CD 9.0 \$29.95

FreeBSD Subscription, start with DVD 9.0 \$29.95

PC-BSD 9.1 DVD (Isotope Edition)

PC-BSD 9.1 DVD \$29.95

PC-BSD Subscription \$19.95

The FreeBSD Handbook

The FreeBSD Handbook, Volume 1 (User Guide) \$39.95

The FreeBSD Handbook, Volume 2 (Admin Guide) \$39.95

The FreeBSD Handbook Specials

The FreeBSD Handbook, Volume 2 (Both Volumes) \$59.95

The FreeBSD Handbook, Both Volumes & FreeBSD 9.1 \$79.95

PC-BSD 9.0 Users Handbook \$24.95

BSD Magazine \$11.99

The FreeBSD Toolkit DVD \$39.95

FreeBSD Mousepad \$10.00

FreeBSD & PCBSD Caps \$20.00

BSD Daemon Horns \$2.00



Bundle Specials!
Save \$\$\$

Just Plain Fun
Mousepads & Novelty Horns



BSD Magazine
Available Monthly



For even MORE items
visit our website today!

www.FreeBSDMall.com

FreeBSD Based Dual-Controller Storage System Concept

by Mikhail E. Zakharov

Nowadays, most of the modern and powerful block-level storage systems around the world are built to work in expensive Fibre Channel or cheaper iSCSI SAN environments. Independent of their class, capacity and performance, they are created on well-known principles, technologies and architectures. Furthermore, some of these systems are using common servers for their controller hardware with Linux or even AIX as storage operating systems on-board.

So theoretically nothing can stop us from developing our own reliable block-level storage system based on FreeBSD, especially keeping in mind existence of successful FreeNAS, iXsystems and NetApp projects. Now let's see what FreeBSD-10.2 could bring us just out of the box, so we do not have to write any additional kernel modules to achieve the goal.

Although our simple model is intended to work on my laptop within the Oracle VirtualBox environment, which is used to run virtual machines for storage controllers along with a client host, we will try to follow the basic principles of reliability, availability, and serviceability (RAS) firstly introduced by IBM. And may the Beastie be with us!

For our experiments, we will need two nearly identical virtual machines (ctrl-a and ctrl-b) for storage controllers and one machine (clnt-1) for a client host. For obvious reasons, we can't test Fibre Channel HBAs on the laptop so we will use iSCSI for our block-level access to the FreeBSD storage system model.

As we don't use Fibre Channel in our environment, the controllers will have 2 LAN connections: one (private) for inter-controller and another (public) for host-controller communications.

ctrl-a	ctrl-b
Private LAN IP-address: 192.168.56.10	Private network IP-address: 192.168.56.11
Public LAN IP-address: 192.168.55.10	Public LAN IP-address: 192.168.55.11

We also need to prepare and configure “shared drives” on both of controllers. This special feature allows us to share physical drives between virtual machines so do not confuse it with “shared folders”, which is a completely different technology. In our case, these “shared drives” will simulate a dummy enclosure with four drives (d0, d1, d2, d3) for our storage system model.

It's better to use a dedicated controller for these shared drives to separate them from the system drive. In the virtual environment it's not very important which one to choose, but it will definitely work with SATA controller emulation. Therefore:

- Using VirtualBox Manager, create a new SATA controller with 4 fixed-sized VDI drives: d0, d1, d2, d3 on ctrl-a machine;
- Then go to VirtualBox Virtual Media Manager and set “shareable” flag for each drive;

Finally, attach drives to the new SATA controller on ctrl-b virtual machine.

Install FreeBSD on ctrl-a and ctrl-b virtual machines using default parameters and ada0 as the drive with root file-system. Then configure general parameters in `/etc/rc.conf`:

ctrl-a	ctrl-b
<pre>hostname="ctrl-a" ifconfig_em0="inet 192.168.56.10 netmask 255.255.255.0" # Inter- controller private network ifconfig_em1="inet 192.168.55.10 netmask 255.255.255.0" # Public network # VirtualBox guest additions vboxguest_enable="YES" vboxservice_enable="YES" # iSCSI ctld_enable="YES" # target iscsid_enable="YES" # initiator</pre>	<pre>hostname="ctrl-b" ifconfig_em0="inet 192.168.56.11 netmask 255.255.255.0" # Inter- controller private network ifconfig_em1="inet 192.168.55.11 netmask 255.255.255.0" # Public network # VirtualBox guest additions vboxguest_enable="YES" vboxservice_enable="YES" # iSCSI ctld_enable="YES" # target iscsid_enable="YES" # initiator</pre>

Now it's important to set iSCSI “disconnection on fail” kernel variable in `/etc/sysctl.conf` on both systems to enable failover to the next controller in case of disaster:

```
kern.iscsi.fail_on_disconnection=1
```

After reboot, according to the `dmesg` output, our shared drives are accessible as `ada1`, `ada2`, `ada3`, `ada4` on both machines:

```
root@ctrl-a:/ # dmesg|grep ada
ada0 at ata0 bus 0 scbus0 target 0 lun 0
```

```
ada0: <VBOX HARDDISK 1.0> ATA-6 device
ada0: Serial Number VB5f1cd9c5-7f39ae5f
ada0: 33.300MB/s transfers (UDMA2, PIO 65536bytes)
ada0: 20480MB (41943040 512 byte sectors: 16H 63S/T 16383C)
ada0: Previously was known as ad0

ada1 at ahcich0 bus 0 scbus2 target 0 lun 0
ada1: <VBOX HARDDISK 1.0> ATA-6 SATA 2.x device
ada1: Serial Number VBa2a70c86-5e4db960
ada1: 300.000MB/s transfers (SATA 2.x, UDMA6, PIO 8192bytes)
ada1: Command Queueing enabled
ada1: 100MB (204800 512 byte sectors: 16H 63S/T 203C)
ada1: Previously was known as ad4

ada2 at ahcich1 bus 0 scbus3 target 0 lun 0
ada2: <VBOX HARDDISK 1.0> ATA-6 SATA 2.x device
ada2: Serial Number VB6a1e2b6c-fcb1fd23
ada2: 300.000MB/s transfers (SATA 2.x, UDMA6, PIO 8192bytes)
ada2: Command Queueing enabled
ada2: 100MB (204800 512 byte sectors: 16H 63S/T 203C)
ada2: Previously was known as ad6

ada3 at ahcich2 bus 0 scbus4 target 0 lun 0
ada3: <VBOX HARDDISK 1.0> ATA-6 SATA 2.x device
ada3: Serial Number VBad8731fa-8e8050c8
ada3: 300.000MB/s transfers (SATA 2.x, UDMA6, PIO 8192bytes)
```



```
ada3: Command Queueing enabled
ada3: 100MB (204800 512 byte sectors: 16H 63S/T 203C)
ada3: Previously was known as ad8
ada4 at ahcich3 bus 0 scbus5 target 0 lun 0
ada4: <VBOX HARDDISK 1.0> ATA-6 SATA 2.x device
ada4: Serial Number VB0f2bbabf-1b8af7ae
ada4: 300.000MB/s transfers (SATA 2.x, UDMA6, PIO 8192bytes)
ada4: Command Queueing enabled
ada4: 100MB (204800 512 byte sectors: 16H 63S/T 203C)
ada4: Previously was known as ad10
```

Now we can create RAIDs on our shared drives. Actually, we may choose any available type (even RAID 0) but as we have four shared drives, let's use GEOM Mirror provider (gmirror) to create two reliable RAID-1 mirrors: one for ctrl-a and another for ctrl-b.

ctrl-a	ctrl-b
root@ctrl-a:/ # gmirror load	root@ctrl-b:/ # gmirror load
root@ctrl-a:/ # gmirror label -v ctrl_a_gm0 /dev/ada1 /dev/ada2	root@ctrl-b:/ # gmirror label -v ctrl_b_gm0 /dev/ada3 /dev/ada4

Note that ada0 is our system drive, so we shouldn't put it under shared RAID configuration.

Then change /boot/loader.conf on both controllers to start gmirror on boot:

ctrl-a	ctrl-b
geom_mirror_load="YES"	geom_mirror_load="YES"

After reboot we can check if gmirror devices are created:

```
root@ctrl-a:/ # ls -la /dev/mirror

total 0

crw-r-----  1 root  operator  0x5d Mar 15 13:44 ctrl_a_gm0
crw-r-----  1 root  operator  0x61 Mar 15 13:44 ctrl_b_gm0
```

The same can be seen on ctrl-b, so I will always skip listings from the other controller to reduce the article size.

Now we can partition the mirrored space. Actually these partitions would be LUNs for our further testing purposes.

Create GPT partition scheme on each mirror:

ctrl-a	ctrl-b
root@ctrl-a:/ # gpart create -s GPT /dev/mirror/ctrl_a_gm0	root@ctrl-b:/ # gpart create -s GPT /dev/mirror/ctrl_b_gm0

To simplify the example we will add only one partition to each mirror:

ctrl-a	ctrl-b
root@ctrl-a:/ # gpart add -t freebsd-ufs -a 1m /dev/mirror/ctrl_a_gm0	root@ctrl-b:/ # gpart add -t freebsd-ufs -a 1m /dev/mirror/ctrl_b_gm0

```
root@ctrl-b:/ # ls -la /dev/mirror/

total 0

crw-r-----  1 root  operator  0x5d Mar 15 13:44 ctrl_a_gm0
crw-r-----  1 root  operator  0x60 Mar 15 13:44 ctrl_a_gm0p1
```

Finally we can check the result:


```
crw-r----- 1 root  operator  0x61 Mar 15 13:44 ctrl_b_gm0
crw-r----- 1 root  operator  0x63 Mar 15 13:44 ctrl_b_gm0p1
```

It's extremely harmful to write data simultaneously from both controllers to the same shared drive. As we don't have any arbitrator yet, we must avoid accessing ctrl_b_gm0 mirror from ctrl-a controller and vice versa: ctrl_a_gm0 mirror from ctrl-b controller. Also we must somehow implement controller redundancy. To achieve both of the goals let's invent this mechanism.

First we have to make links from one controller to another to reach the opposite mirror. We can try to do it with ggate (GEOM gate provider) or by utilizing iSCSI. I chose iSCSI for our testing because it looks more flexible and therefore suitable for our purposes.

At this point we will have two paths on each controller to the opposite controller mirror: one path is local, as we can access physical drives directly, another is provided by iSCSI connection. Therefore, our second step would be to create gmultipath (GEOM Multipath) device for both of the links.

This construction will help us to redirect I/O from a failed controller to an active one. Nothing seems impossible for now, so let's create this arbitrator construction for our LUN partitions.

We don't need to enable iSCSI -target and -initiator explicitly, as we already put appropriate variables to /etc/rc.conf of both controllers. Therefore, we will just create iSCSI-target configuration files /etc/ctl.conf:

/etc/ctl.conf on ctrl-a	/etc/ctl.conf on ctrl-b
<pre>portal-group pg0 { discovery-auth-group no- authentication listen 192.168.56.10 }</pre>	<pre>portal-group pg0 { discovery-auth-group no- authentication listen 192.168.56.11 }</pre>

/etc/ctl.conf on ctrl-a	/etc/ctl.conf on ctrl-b
<pre>target iqn. 2016-01.local.sss.private:target0 { auth-group no- authentication portal-group pg0 lun 0 { path /dev/mirror/ ctrl_a_gm0p1 size 102760448 } }</pre>	<pre>target iqn. 2016-01.local.sss.private:target0 { auth-group no- authentication portal-group pg0 lun 0 { path /dev/mirror/ ctrl_b_gm0p1 size 102760448 } }</pre>

As you can see, we will use Portal group pg0 for our private inter-controller communication. The size of the LUN 0 is 102760448 bytes. This value is taken from the output of:

```
root@ctrl-a:/ # gpart list mirror/ctrl_a_gm0

Geom name: mirror/ctrl_a_gm0
modified: false
state: OK
fwheads: 32
fwsectors: 9
last: 204765
first: 34
```



```
entries: 128
```

```
scheme: GPT
```

Providers:

```
Mediasize: 102760448 (98M)
```

```
Sectorsize: 512
```

```
Stripesize: 0
```

```
Stripeoffset: 1048576
```

```
Mode: r0w0e0
```

```
rawuuid: 5677c154-e917-11e5-944d-080027868477
```

```
rawtype: 516e7cb6-6ecf-11d6-8ff8-00022d09712b
```

```
label: 1
```

```
length: 102760448
```

```
offset: 1048576
```

```
type: freebsd-ufs
```

```
index: 1
```

```
end: 202751
```

```
start: 2048
```

Consumers:

```
1. Name: mirror/ctrl_a_gm0
```

```
Mediasize: 104857088 (100M)
```

```
Sectorsize: 512
```

```
Mode: r0w0e0
```

Check Mediasize value of mirror/ctrl_a_gm0p1 provider. The Mediasize of mirror/ctrl_b_gm0 must match the result.

Now we can start iSCSI -target (ctld) on both controllers and try to reach drives:

ctrl-a	ctrl-b
root@ctrl-a:/ # /etc/rc.d/ctld start Starting ctld.	oot@ctrl-b:/ # /etc/rc.d/ctld start Starting ctld.
root@ctrl-b:/ # iscsictl -A -p 192.168.56.11 -t iqn. 2016-01.local.sss.private:target0	root@ctrl-b:/ # iscsictl -A -p 192.168.56.10 -t iqn. 2016-01.local.sss.private:target0

After it, the dmesg command will show us that a new da0 drive has appeared on each controller:

```
root@ctrl-a:/ # dmesg| tail -6

da0 at iscsi1 bus 0 scbus7 target 0 lun 0

da0: <FREEBSD CTLDISK 0001> Fixed Direct Access SPC-4 SCSI device
da0: Serial Number MYSERIAL    0
da0: 150.000MB/s transfers
da0: Command Queueing enabled
da0: 98MB (200704 512 byte sectors: 64H 32S/T 98C)
FBSDprivate_1.0 {
...
    _custom_func;
        __sys_custom_func;
...
}
```

Now let's save this connection's definitions in the /etc/iscsi.conf file to refer to it later:

ctrl-a	ctrl-b
<pre>private { TargetAddress = 192.168.56.11 TargetName = iqn. 2016-01.local.sss.private:target0 }</pre>	<pre>private { TargetAddress = 192.168.56.10 TargetName = iqn. 2016-01.local.sss.private:target0 }</pre>

Now that the first part of the arbitrator construction is done, let's provide it with a failover ability:

ctrl-a	ctrl-b
<pre>root@ctrl-a:/ # gmultipath create CTRL_B_BACK /dev/da0 /dev/mirror/ ctrl_b_gm0p1</pre>	<pre>root@ctrl-b:/ # gmultipath create CTRL_A_BACK /dev/da0 /dev/mirror/ ctrl_a_gm0p1</pre>

Now pay attention that we use “manual” method for multipathing device creation. In this case, multipathing configuration is not stored in the metadata area of the device. And although this information will certainly be lost on reboot, we gain some profit on iSCSI configuration for the client-host.

Check the result:

ctrl-a	ctrl-b
<pre>root@ctrl-a:/ # dmesg tail -4 GEOM_MULTIPATH: CTRL_B_BACK created GEOM_MULTIPATH: da0 added to CTRL_B_BACK GEOM_MULTIPATH: da0 is now active path in CTRL_B_BACK GEOM_MULTIPATH: mirror/ ctrl_b_gm0p1 added to CTRL_B_BACK</pre>	<pre>root@ctrl-b:/ # dmesg tail -4 GEOM_MULTIPATH: CTRL_A_BACK created GEOM_MULTIPATH: da0 added to CTRL_A_BACK GEOM_MULTIPATH: da0 is now active path in CTRL_A_BACK GEOM_MULTIPATH: mirror/ ctrl_a_gm0p1 added to CTRL_A_BACK</pre>

We have created multipathing with active-passive policy, and active path is `/dev/da0` now, which means the opposite controller. So the data will always be routed through the opposite controller until it fails. This simple mechanism should prevent data corruption we mentioned above.

Now we are ready to provide storage to the client host `clnt-1`. Therefore we have to update iSCSI target configuration file `/etc/ctld.conf` on controllers with “public” definitions:

ctrl-a	ctrl-b
<pre>portal-group pg0 { discovery-auth-group no- authentication listen 192.168.56.10 } portal-group pg1 { discovery-auth-group no- authentication listen 192.168.55.10 } # Private - inter-node LUN target iqn. 2016-01.local.sss.private:target0 { auth-group no- authentication portal-group pg0</pre>	<pre>portal-group pg0 { discovery-auth-group no- authentication listen 192.168.56.11 } portal-group pg1 { discovery-auth-group no- authentication listen 192.168.55.11 } # Private - inter-node LUN target iqn. 2016-01.local.sss.private:target0 { auth-group no- authentication portal-group pg0</pre>

ctrl-a	ctrl-b
<pre> lun 0 { path /dev/mirror/ ctrl_a_gm0p1 size 102760448 } } # Public - client access LUNs target iqn. 2016-01.local.sss.public:target0 { auth-group no- authentication portal-group pg1 # Direct path to the local LUN lun 0 { path /dev/mirror/ ctrl_a_gm0p1 size 102760448 } # LUN owned by the opposite controller lun 1 { path /dev/ </pre>	<pre> lun 0 { path /dev/mirror/ ctrl_b_gm0p1 size 102760448 } } # Public - client access LUNs target iqn. 2016-01.local.sss.public:target0 { auth-group no- authentication portal-group pg1 # Direct path to the local LUN lun 0 { path /dev/mirror/ ctrl_b_gm0p1 size 102760448 } # LUN owned by the opposite controller lun 1 { path /dev/ </pre>

ctrl-a	ctrl-b
<pre>multipath/CTRL_B_BACK size 102760448 } }</pre>	<pre>multipath/CTRL_A_BACK size 102760448 } }</pre>

We add Port group pg1 for public host connection. Also, we export two LUNs from each controller to the client host. Ergo on each controller LUN 0 is provided through so called owner controller, and LUN 1 is the partition from the opposite controller provided through the arbitrator construction.

Now ctld daemon should be forced to re-read its configuration file. On both controllers run:

```
killall -1 ctld
```

Controllers of the storage system are ready now to serve iSCSI requests, therefore, we can leave them for a while and configure the client host. We need to add following lines to the `/etc/rc.conf` file to setup its hostname, network parameters and iSCSI initiator:

```
hostname="clnt-1"

ifconfig_em0="inet 192.168.55.20 netmask 255.255.255.0"

# VirtualBox guest additions
vboxguest_enable="YES"
vboxservice_enable="YES"

iscsid_enable="YES"           # SCSI initiator
```


Then set required iSCSI kernel variable via `/etc/sysctl.conf`:

```
kern.iscsi.fail_on_disconnection=1
```

After the reboot, we can start our storage related tasks on the client side. Let's access iSCSI targets on both controllers.

For `ctrl-a` and `ctrl-b` run:

```
root@clnt-1:/ # iscsictl -A -p 192.168.55.10 -t
iqn.2016-01.local.sss.public:target0

root@clnt-1:/ # iscsictl -A -p 192.168.55.11 -t
iqn.2016-01.local.sss.public:target0
```

Then check the result:

```
root@clnt-1:/ # iscsictl -L
```

Target name	Target portal	State
iqn.2016-01.local.sss.public:target0	192.168.55.10	Connected: da0 da1
iqn.2016-01.local.sss.public:target0	192.168.55.11	Connected: da2 da3

Also, it's useful to see the `dmesg` output:

```
da0 at iscsi4 bus 0 scbus2 target 0 lun 0
da0: <FREEBSD CTLDISK 0001> Fixed Direct Access SPC-4 SCSI device
da0: Serial Number MYSERIAL 1
da0: 150.000MB/s transfers
da0: Command Queueing enabled
da1: 150.000MB/s transfers
```

```
da0: Command Queueing enabled

da1: 150.000MB/s transfers

da0: 98MB (200704 512 byte sectors: 64H 32S/T 98C)

da1 at iscsi4 bus 0 scbus2 target 0 lun 1

da1: <FREEBSD CTLDISK 0001> Fixed Direct Access SPC-4 SCSI device

da1: Serial Number MYSERIAL    2

da1: Command Queueing enabled

da1: 98MB (200704 512 byte sectors: 64H 32S/T 98C)

da2 at iscsi5 bus 0 scbus3 target 0 lun 0

da2: <FREEBSD CTLDISK 0001> Fixed Direct Access SPC-4 SCSI device

da2: Serial Number MYSERIAL    1

da2: 150.000MB/s transfers

da2: Command Queueing enabled

da2: 98MB (200704 512 byte sectors: 64H 32S/T 98C)

da3 at iscsi5 bus 0 scbus3 target 0 lun 1

da3: <FREEBSD CTLDISK 0001> Fixed Direct Access SPC-4 SCSI device

da3: Serial Number MYSERIAL    2

da3: 150.000MB/s transfers

da3: Command Queueing enabled

da3: 98MB (200704 512 byte sectors: 64H 32S/T 98C)
```

Everything looks fine.

Keep in mind that da1 drive on ctrl-a is our arbitrator mechanism, which points now to the partition p1 on the RAID-1 ctrl_b_gm0 mirror, which actually resides on ctrl-b. Similarly, da3 provided by ctrl-b is the arbitrator construction pointing now to the partition p1 on the RAID-1 ctrl_a_gm0 mirror, which resides on ctrl-a.

These paths are routed through opposite, non-owner controllers. We also have short, direct paths provided by owner controllers: da0 is `ctrl_a_gm0p1` provided by `ctrl-a` and da2 is `ctrl_b_gm0p1` accessible through `ctrl-b`.

The Figure 1 shows our storage system architecture layout we just created.

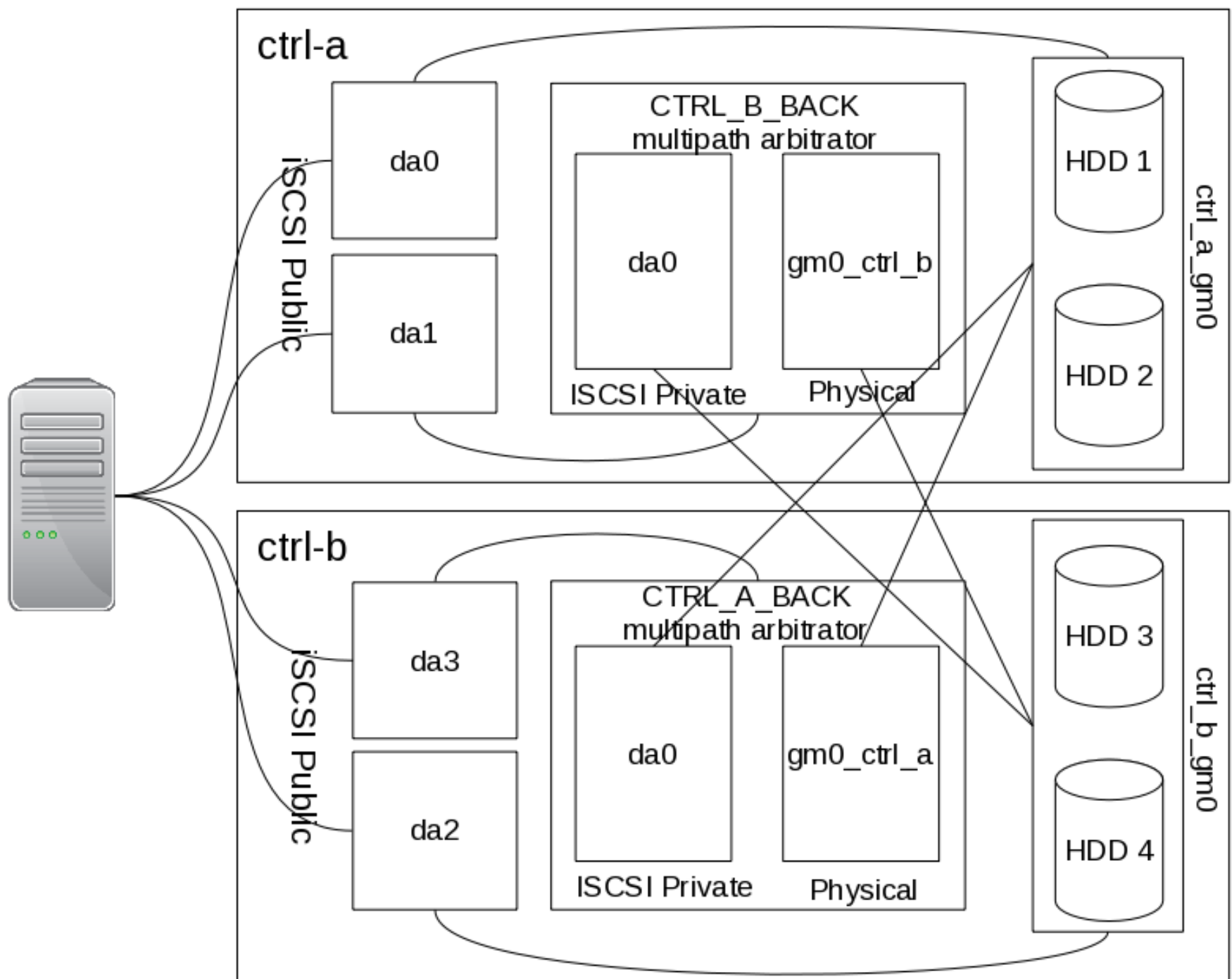


Figure 1. Dual-controller storage architecture overview.

As we have 2 paths for each LUN on our clnt-1 host, we can enable multipathing:

```
root@clnt-1:/ # gmultipath create CTRL_A /dev/da0 /dev/da3
root@clnt-1:/ # gmultipath create CTRL_B /dev/da2 /dev/da1
```


We can check the result with the `gmultipath list` command or by reading `dmesg` output:

```
root@clnt-1:/ # dmesg | grep GEOM_MULTIPATH
GEOM_MULTIPATH: CTRL_A created
GEOM_MULTIPATH: da0 added to CTRL_A
GEOM_MULTIPATH: da0 is now active path in CTRL_A
GEOM_MULTIPATH: da3 added to CTRL_A
GEOM_MULTIPATH: CTRL_B created
GEOM_MULTIPATH: da2 added to CTRL_B
GEOM_MULTIPATH: da2 is now active path in CTRL_B
GEOM_MULTIPATH: da1 added to CTRL_B
```

This storage architecture with our simple arbitrator and absence of cache is asymmetric by nature so it has to run in active-passive mode where `da0` and `da2` are active paths now. Sure we can enable active-active or active-read modes to use all four paths and everything will work without errors. But it will cause potential overheads as the data for every LUN will also be routed through the LAN to non-owner controllers.

Now we can do whatever is needed to use drives on our client. For example, let's label them and create filesystems on the multipathed devices:

```
root@clnt-1:/ # disklabel -Brw /dev/multipath/CTRL_A auto
root@clnt-1:/ # disklabel -Brw /dev/multipath/CTRL_B auto
root@clnt-1:/ # newfs /dev/multipath/CTRL_A
/dev/multipath/CTRL_A: 98.0MB (200704 sectors) block size 32768, frag-
ment size 4096
    using 4 cylinder groups of 24.53MB, 785 blks, 3200 inodes.
super-block backups (for fsck_ffs -b #) at:
    192, 50432, 100672, 150912
```

```
root@clnt-1:/ # newfs /dev/multipath/CTRL_B

/dev/multipath/CTRL_B: 98.0MB (200704 sectors) block size 32768, frag-
ment size 4096

    using 4 cylinder groups of 24.53MB, 785 blks, 3200 inodes.

super-block backups (for fsck_ffs -b #) at:
    192, 50432, 100672, 150912
```

Finally, create mount points and mount filesystems:

```
root@clnt-1:/ # mkdir -p /storage/d0 /storage/d1

root@clnt-1:/ # mount /dev/multipath/CTRL_A /storage/d0

root@clnt-1:/ # mount /dev/multipath/CTRL_B /storage/d1
```

We are ready now to make various tests and try to break the whole construction to check our architecture. For example, forcibly power-off one of the controllers during a file copy process:

```
root@clnt-1:/ # ls -la ports.tgz

-rw-r--r--  1 root  zmey  59546274 Mar  9 16:22 ports.tgz

root@clnt-1:/ # cp ports.tgz /storage/d0 & cp ports.tgz /storage/d1

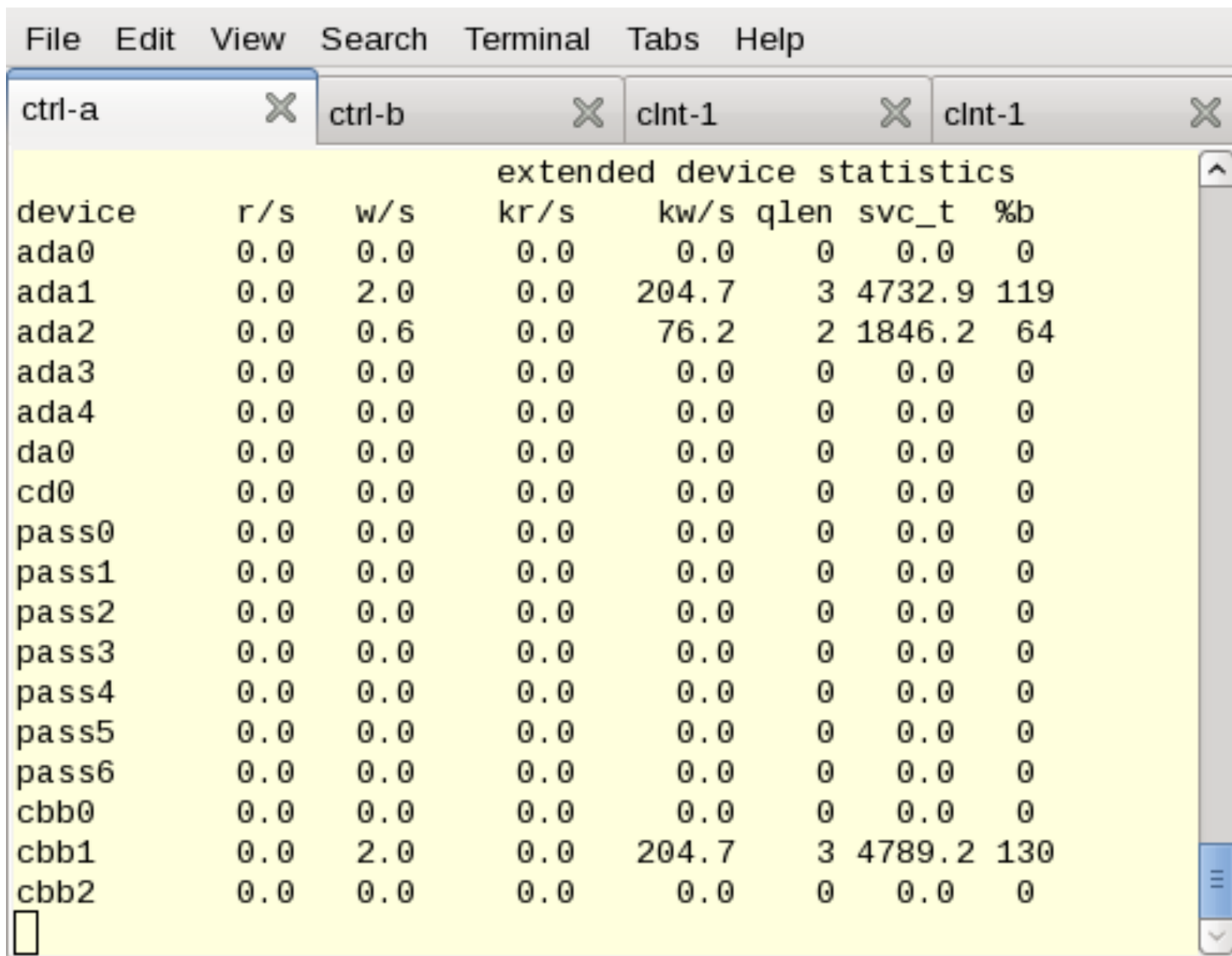
[1] 1268
```

In the next figures, we will see performance statistics from each controller and a client. The values are extremely poor as VirtualBox shared drives were intentionally put on slow USB memory sticks in order to see the whole process in detail.

Figure 2 shows ctrl-a controller and we can see that ada1 and ada2 drives have workload. This is our `ctrl_a_gm0 RAID-1 mirror`. Note we don't see any workload on ada3/ada4 as these drives are parts of `ctrl_b_gm0 mirror` and therefore are governed by ctrl-b.

The working cbb1 device looks like iSCSI target for public LAN `clnt-1` connection, and `da0`, which has zero values of workload, is the link to the LUN on the opposite controller.

We can state that the traffic goes normally through the owner controller, and our arbitrator works well.



extended device statistics							
device	r/s	w/s	kr/s	kw/s	qlen	svc_t	%b
ada0	0.0	0.0	0.0	0.0	0	0.0	0
ada1	0.0	2.0	0.0	204.7	3	4732.9	119
ada2	0.0	0.6	0.0	76.2	2	1846.2	64
ada3	0.0	0.0	0.0	0.0	0	0.0	0
ada4	0.0	0.0	0.0	0.0	0	0.0	0
da0	0.0	0.0	0.0	0.0	0	0.0	0
cd0	0.0	0.0	0.0	0.0	0	0.0	0
pass0	0.0	0.0	0.0	0.0	0	0.0	0
pass1	0.0	0.0	0.0	0.0	0	0.0	0
pass2	0.0	0.0	0.0	0.0	0	0.0	0
pass3	0.0	0.0	0.0	0.0	0	0.0	0
pass4	0.0	0.0	0.0	0.0	0	0.0	0
pass5	0.0	0.0	0.0	0.0	0	0.0	0
pass6	0.0	0.0	0.0	0.0	0	0.0	0
cbb0	0.0	0.0	0.0	0.0	0	0.0	0
cbb1	0.0	2.0	0.0	204.7	3	4789.2	130
cbb2	0.0	0.0	0.0	0.0	0	0.0	0

Figure 2. Normal work of the ctrl-a controller

Figure 3 shows the similar picture on `ctrl-b`, the only exception is that `ada3` and `ada4` are parts of `ctrl_b_gm0` mirror. As we can see, the controller `ctrl-b` works normally and the traffic is directed through the shortest path to the `ctrl_b_gm0` mirror.

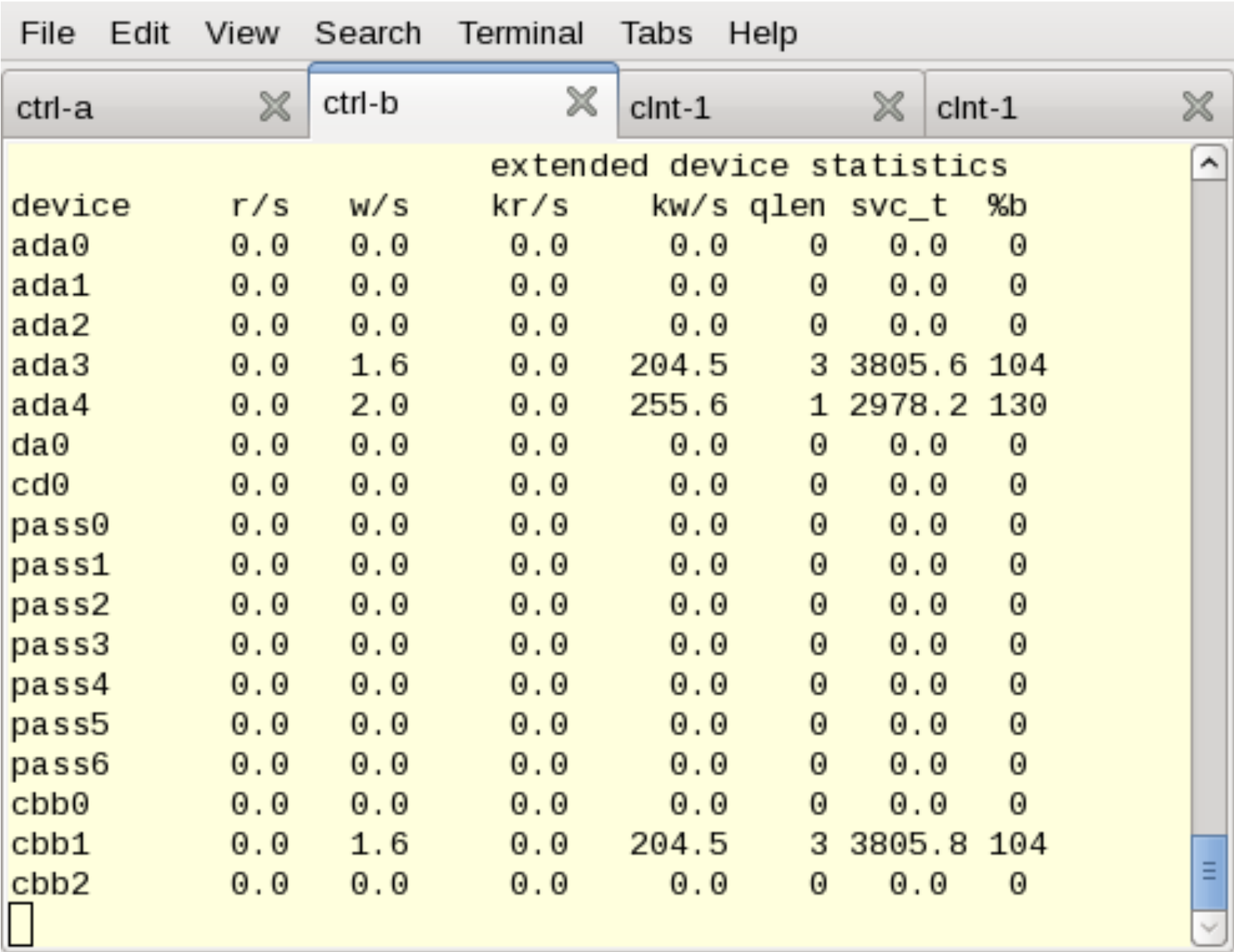


Figure 3. Normal work of the ctrl-b controller

Figure 4 reflects the situation on the clnt-1 client. We can see da0 and da2 are utilized. These devices are primary, active and the shortest paths to the backend RAID-1 mirror drives. Paths da1 and da3 are not loaded according to the active-passive mode of CTRL_A and CTRL_B multipathing devices.

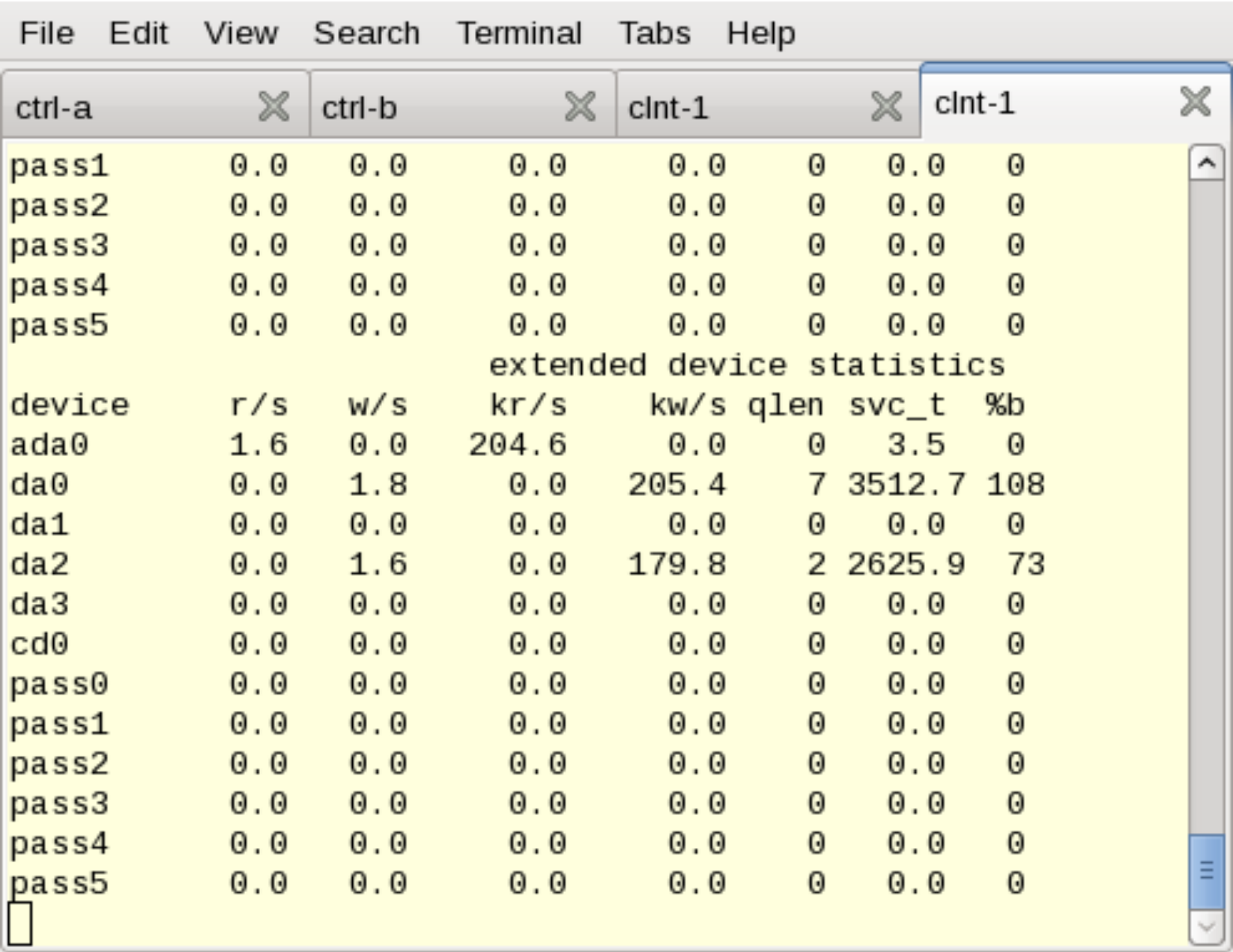


Figure 4. Normal work of the clnt-1.

Now let's fail one of the controllers and check if the system survives it. We can forcible power-off or reset ctrl-a, for example.

Figure 5 shows the reaction of the client. We can see some mess in the performance statistics resulted from failed paths.

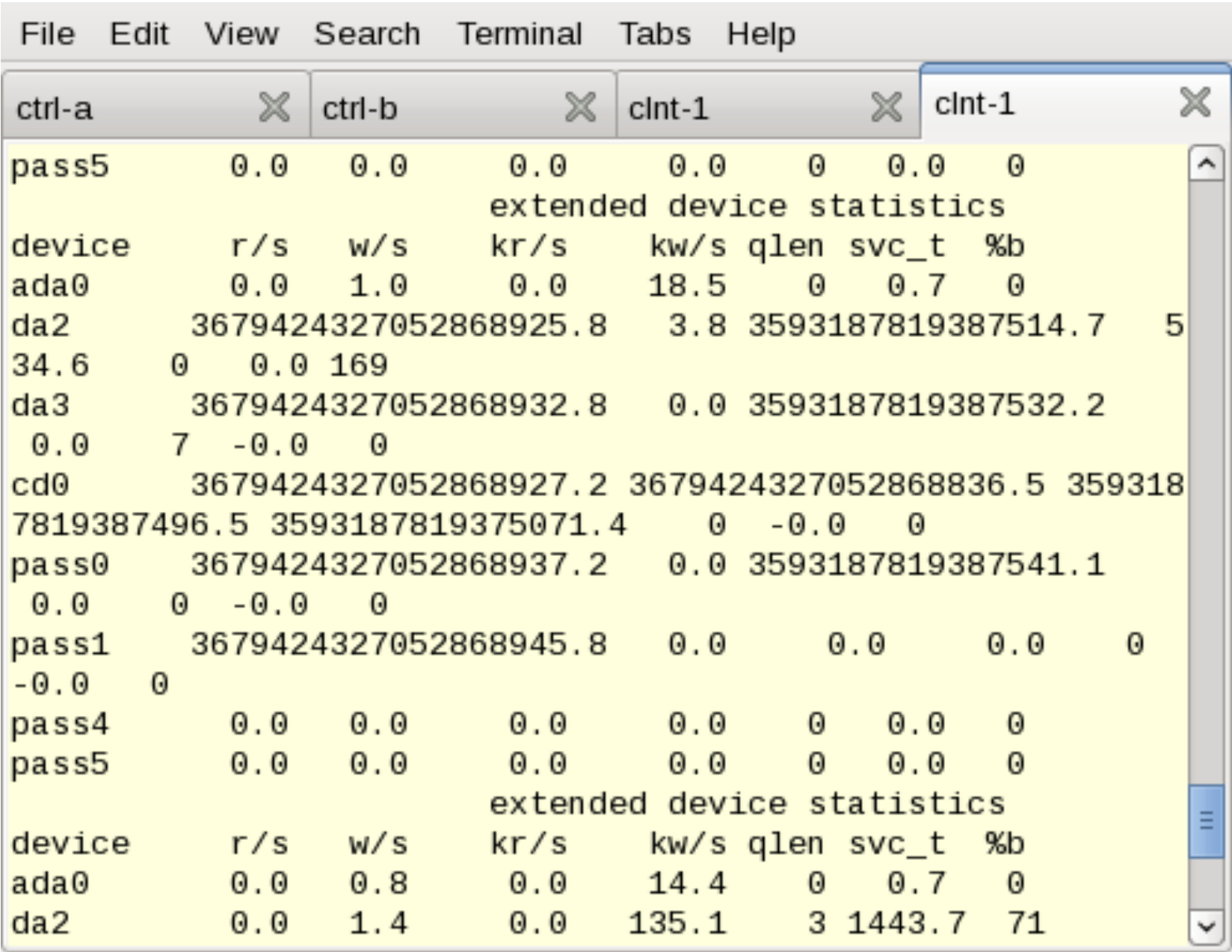


Figure 5. The clnt-1 host immediately after the ctrl-a failure.

Soon the situation stabilizes. Caused by the multipath policy, workload successfully moved from failed da0 to da3 device. This can be seen in Figure 6:

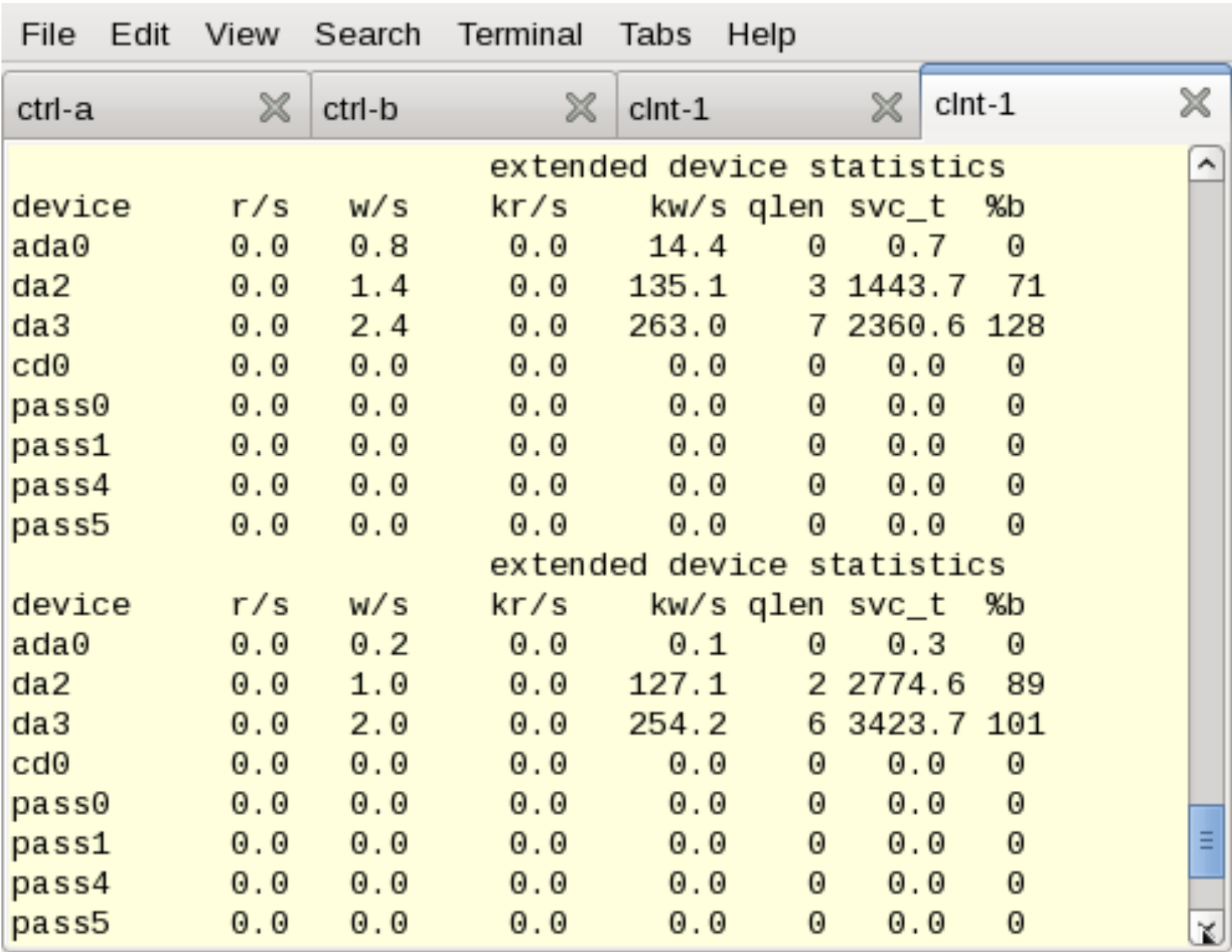


Figure 6. The clnt-1 operates now with the ctrl-b only.

Let's see what is going on the ctrl-b. Figure 7 shows us that ctrl-b now takes the full workload for both LUNs (cbb1/cbb2). Both mirrors ctrl_a_gm0 (ada1/ada2) and ctrl_b_gm0 (ada3/ada4) are also utilized.

File Edit View Search Terminal Tabs Help

ctrl-a X ctrl-b X clnt-1 X clnt-1 X

cbb1	0.0	1.4	0.0	153.9	3	2340.9	111
cbb2	0.0	2.2	0.0	256.0	3	3191.0	106
extended device statistics							
device	r/s	w/s	kr/s	kw/s	qlen	svc_t	%b
ada0	0.0	0.2	0.0	0.8	0	0.2	0
ada1	0.0	0.8	0.0	102.2	11	2219.7	80
ada2	0.0	1.4	0.0	129.4	9	1714.0	112
ada3	0.0	2.0	0.0	255.6	0	1688.9	114
ada4	0.0	2.0	0.0	255.6	0	1177.5	94
cd0	0.0	0.0	0.0	0.0	0	0.0	0
pass0	0.0	0.0	0.0	0.0	0	0.0	0
pass1	0.0	0.0	0.0	0.0	0	0.0	0
pass2	0.0	0.0	0.0	0.0	0	0.0	0
pass3	0.0	0.0	0.0	0.0	0	0.0	0
pass4	0.0	0.0	0.0	0.0	0	0.0	0
pass5	0.0	0.0	0.0	0.0	0	0.0	0
cbb0	0.0	0.0	0.0	0.0	0	0.0	0
cbb1	0.0	2.0	0.0	255.6	0	1708.9	118
cbb2	0.0	1.0	0.0	127.8	11	2205.7	81

Figure 7. The ctrl-b workload after the ctrl-a failure.

After files have been copied, we can check if the data were written correctly:

```
root@cln-1:/ # md5 /storage/d0/ports.tgz
MD5 (/storage/d0/ports.tgz) = 82a5d6a7a3a89b7a5185a543fa6b3a56
root@cln-1:/ # md5 /storage/d1/ports.tgz
MD5 (/storage/d1/ports.tgz) = 82a5d6a7a3a89b7a5185a543fa6b3a56
root@cln-1:/ # md5 ports.tgz
MD5 (ports.tgz) = 82a5d6a7a3a89b7a5185a543fa6b3a56
```

MD5 sums are equal on both remote LUNs and on the local filesystem, so everything looks fine. Our simple storage has successfully survived one controller failure and we can say that our mission is completed at this stage.

Instead of creating filesystem and mounting separate partitions on /storage/d0 and /storage/d1 in the example above, we could do an even more real life storage-like thing: make a stripe of two partitions on the client host:

```
root@clnt-1:/ # gstripe create -v -s 131072 STORAGE /dev/multipath/
CTRL_A /dev/multipath/CTRL_B

root@clnt-1:/ # newfs /dev/stripe/STORAGE

/dev/stripe/STORAGE: 196.0MB (401408 sectors) block size 32768, frag-
ment size 4096

    using 4 cylinder groups of 49.03MB, 1569 blks, 6400 inodes.

super-block backups (for fsck_ffs -b #) at:
    192, 100608, 201024, 301440

root@clnt-1:/ # mount /dev/stripe/STORAGE /storage/d0
```

Then copy files:

```
root@clnt-1:/ # cp ports.tgz /storage/d0/ports.tgz & cp ports.tgz
/storage/d0/ports.1.tgz
```

Then copy files:

```
root@clnt-1:/ # cp ports.tgz /storage/d0/ports.tgz & cp ports.tgz  
/storage/d0/ports.1.tgz
```

Finally, power-off one of the controllers and check the result. It will be the same as in the previous experiment:

```
root@clnt-1:/ # md5 ports.tgz  
  
MD5 (ports.tgz) = 82a5d6a7a3a89b7a5185a543fa6b3a56  
  
root@clnt-1:/ # md5 /storage/d0/ports.tgz  
  
MD5 (/storage/d0/ports.tgz) = 82a5d6a7a3a89b7a5185a543fa6b3a56  
  
root@clnt-1:/ # md5 /storage/d0/ports.1.tgz  
  
MD5 (/storage/d0/ports.1.tgz) = 82a5d6a7a3a89b7a5185a543fa6b3a56  
  
root@v4_cln_1:/home/zmey # cp ports.tgz /storage/d0/
```

In other words, our high-available model is working well.

But to bring the controller back into business we need to do a lot of manual work to reconstruct it after the failure: carefully re-enable iSCSI connections between controllers, restore lost multipath devices and set active paths, finally resume iSCSI and multipath connections to the client host.

During my experiments, I had to reboot controllers hundreds of times. It was really tedious to do everything manually so I wrote a simple shell-script to restore the original storage configuration after a clean boot of both controllers, though in case of a failure I still have to do everything manually.

This script uses my lightweight expect-like tool called “empty,” to automate interactive login and su operations. If you decide to use this script, don't forget to install “empty” from ports or packages. But beware of the script insecurity as it contains plain text passwords right in the body. Don't do such things in production environments, use SSH with keys instead.

The script must be run from the client host by root:

```
#!/bin/sh

login=storadmin

ctrl_a=192.168.55.10

ctrl_b=192.168.55.11

passwd=TopSecret

root_passwd=VeryTopSecret


# On controller A

empty -f ssh $login@$ctrl_a

empty -w -t 5 assword "$passwd\n"

empty -s "su\n"

empty -w -t 5 assword "$root_passwd\n"


empty -s "gmultipath create CTRL_B_BACK /dev/mirror/ctrl_b_gm0p1\n"

empty -s "/etc/rc.d/ctld onestart\n"

empty -s "exit\n"

empty -s "exit\n"


sleep 5


# On controller B

empty -f ssh $login@$ctrl_b
```



```
empty -w -t 5 assword "$passwd\n"

empty -s "su\n"

empty -w -t 5 assword "$root_passwd\n"


empty -s "gmultipath create CTRL_A_BACK /dev/mirror/ctrl_a_gm0p1\n"
empty -s "/etc/rc.d/ctld onestart\n"
empty -s "iscsictl -A -n private\n"
empty -s "gmultipath add CTRL_A_BACK /dev/da0\n"
empty -s "gmultipath rotate CTRL_A_BACK\n"
empty -s "exit\n"
empty -s "exit\n"


sleep 5

# On controller A

empty -f ssh $login@$ctrl_a

empty -w -t 5 assword "$passwd\n"

empty -s "su\n"

empty -w -t 5 assword "$root_passwd\n"


empty -s "iscsictl -A -n private\n"
empty -s "gmultipath add CTRL_B_BACK /dev/da0\n"
empty -s "gmultipath rotate CTRL_B_BACK\n"
empty -s "exit\n"
```

```
empty -s "exit\n"

# On the client. Common part

iscsictl -A -p 192.168.55.10 -t iqn.2016-01.local.sss.public:target0
iscsictl -A -p 192.168.55.11 -t iqn.2016-01.local.sss.public:target0

gmultipath create CTRL_A /dev/da0 /dev/da3
gmultipath create CTRL_B /dev/da2 /dev/da1

# On the client. Test 1

newfs /dev/multipath/CTRL_A
newfs /dev/multipath/CTRL_A
mount /dev/multipath/CTRL_A /storage/d0
mount /dev/multipath/CTRL_B /storage/d1

# On the client. Test 2

# gstripe create -v -s 131072 STORAGE /dev/multipath/CTRL_A /dev/
multipath/CTRL_B

# newfs /dev/stripe/STORAGE

# mount /dev/stripe/STORAGE /storage/d0
```

Real storage systems have serious clustering software for the management and synchronization of controllers, but in our case it will be in the future.

Actually, we have a lot of directions for the development from this point. We didn't experiment with ZFS yet, didn't implement mirrored data cache (which is the main part of any modern storage system), and we didn't even test our architecture on physical hardware and Fibre Channel host bus adapters.

Our system reliably works, but honestly, this is only a proof of concept which must be seriously tested before being used outside the laboratory.



About the Author:

My name is Mikhail E. Zakharov and I am a proud SAN/storage IBMer. 10 years of experience in large SAN and storage environments: mainly Hitachi, HP and Brocade. Empty – expect-like tool author. FreeBSD enthusiast.

Benchmarks Mac OS X vs FreeBSD

by Natalia Portillo

Imagine you've got an old Macintosh sitting around and you think you may be able to use it for other purposes. You have some expertise with FreeBSD and feel quite confident using it. In this article, I'll try to explore what's better, to install Mac OS X until Apple stops supporting your exact model, or move on to FreeBSD. For this purpose, I'll explore several advantages and disadvantages for diverse use case scenarios as well as pure benchmarking for both.

The hardware used for benchmarks was an Apple MacBook Pro Mid 2009 with a Core 2 Duo @ 2.66Ghz, and a Samsung 840Pro 256Gb SSD and 8Gb of DDR3-1066 RAM.

Advantages of Mac OS X:

- All hardware is supported out of the box, no need for wireless firmware configuration.
- For some usage cases scenarios that require closed-source or commercial software, that software is not available for FreeBSD.
- Supports, if you need it, the proprietary Apple ecosystem: iCloud, iTunes, AirPlay, etc.
- If you have a SOHO with a lot of Macintosh computers, or a lot of iOS devices, Mac OS X Server provides services that you cannot find anywhere else (local iCloud mirror, Apple Store mirror, login services, etc...)

Advantages of FreeBSD:

- Open source software.

- The community is more problem-solving, you can find solutions more easily, while on Mac OS X communities a lot of solutions depend on Apple.
- You can completely disable the GUI for headless systems, while on Mac OS X it will always be running, taking up memory and cycles.
- You can migrate it to another computer, Macintosh or not, as simply as moving the disk it is installed on. Mac OS X supports that only partially (newer models don't support older versions) and only between Macintosh computers.
- While Macintosh hardware may need manual configuration, external peripherals (like TV tuners, SSDs, USB gadgets, etc.) have usually better support in FreeBSD.

Filesystems benchmarks and comparison:

When talking about Mac OS X and FreeBSD, there are right now only three filesystems to take into consideration: UFS2, HFS+ and ZFS.

UFS (Unix File System):

UFS comes from 4.2BSD (where it was called Fast File System). It is a very simple filesystem using tables for directories and inodes for files. It is supported by almost all Unixes, including Mac OS X up to version 10.5, however, the old versions had some incompatible features that prevented disk interchange between them. Nowadays, it has been extended as UFS2 in FreeBSD adding support for extents, extended attributes, ACLs and journaling. UFS2 supports disk interchange between modern BSD flavors (FreeBSD, NetBSD, OpenBSD and DragonFly BSD). It is also a filesystem that preserves and enforces filename case (that is “foo” and “Foo” are different files).

HFS+ (Hierarchical File System Plus):

HFS+, also called Mac OS Extended, is the native filesystem for Mac OS X, the natural evolution for the older HFS filesystem present since Mac OS 2.1. It is also a simple filesystem that uses a single B-Tree for directories and files. Support for journaling, extended attributes, in-line compression and ACLs has all been added in different Mac OS X version. It is by default a filename case preserving but insensitive filesystem, but can be initialized as a filename case sensitive one (called HFSX).

Filesystem encryption and redundancy:

Neither of the native filesystems of FreeBSD and Mac OS X support filesystem encryption or redundancy. However, FreeBSD provides GEOM that gives redundancy (as RAID levels), concatenation and encryption (GELI) of disks that can be used with any supported filesystem,

while Mac OS X provides AppleRAID for the redundancy and concatenation and CoreStorage for the encryption and hybrid disk caching (having an SSD cache over a traditional HDD storage).

ZFS (Zettabyte File System):

Probably one of most advanced filesystems, introduced by Sun in the Solaris operating system, open sourced, and then closed by Oracle, has been ported to almost all other operating systems, including Mac OS X and FreeBSD (where it is to completely substitute for UFS in the near future). It is a complex filesystem, offering all the features plus more. It supports extended attributes, copy-on-write, ACLs, does not need journaling by design, detects silent corruption of data, allows for redundancy and concatenation without an extra layer behind, supports in-line compression and encryption (this is only on Solaris closed-source version), can be initialized as case insensitive or sensitive, read-only and write-only caching on another disk (e.g. SSD), deduplication, snapshots, and more. All of the features added by Oracle since it closed the source are not available outside Solaris, but all new features added to the open source version are available on all operating systems supported, thanks to the OpenZFS efforts.

Benchmarks:

All benchmarks have been done on the same hardware, using a separate partition from the system, with everything already loaded, a GUI loaded and user logged-in, indexing services stopped (Spotlight on Mac OS X and Baloo on FreeBSD), and TRIM enabled. The whole partition was TRIM-ed and formatted between tests. HFS+ and ZFS on Mac OS X have been initialized as case insensitive, while UFS2 and ZFS on FreeBSD have been initialized as case sensitive, the default behavior. Changing case sensitiveness has not demonstrated any measurable difference in performance. Benchmark software used was Bonnie++ 1.97 with the following command line:

```
bonnie++ -s 16G -n 1000 -r 8192 -f -b.
```

Sequential write

In this test, Bonnie++ creates a few files and writes to them sequentially, measuring the throughput.

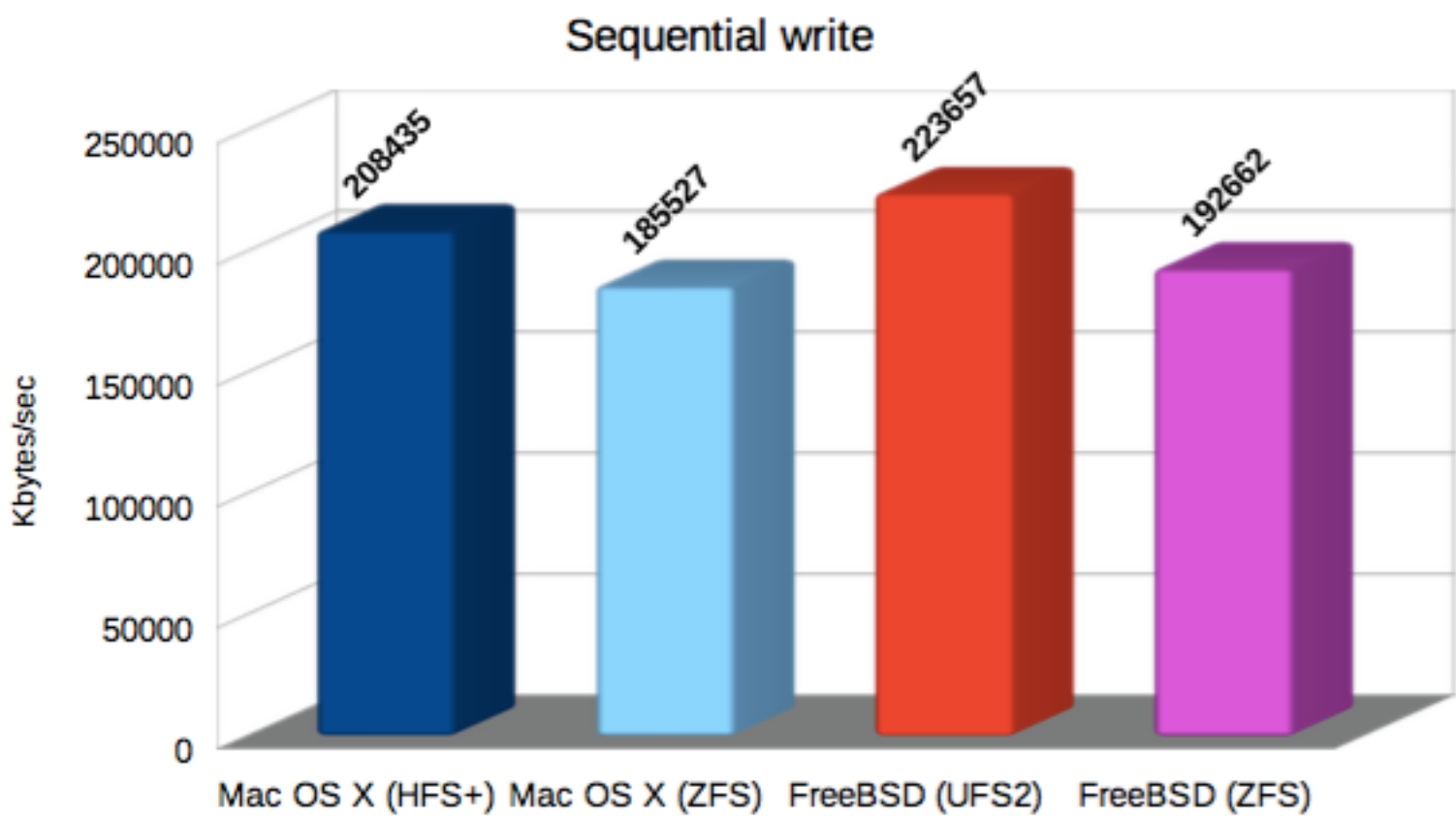


Figure 1. Sequential write.

FreeBSD shows a small but not significant advantage (less than 10%) over Mac OS X, with ZFS behaving slower than the native filesystem on both operating systems.

Sequential rewrite

In this test, Bonnie++ rewrites the files it previously created with new data.

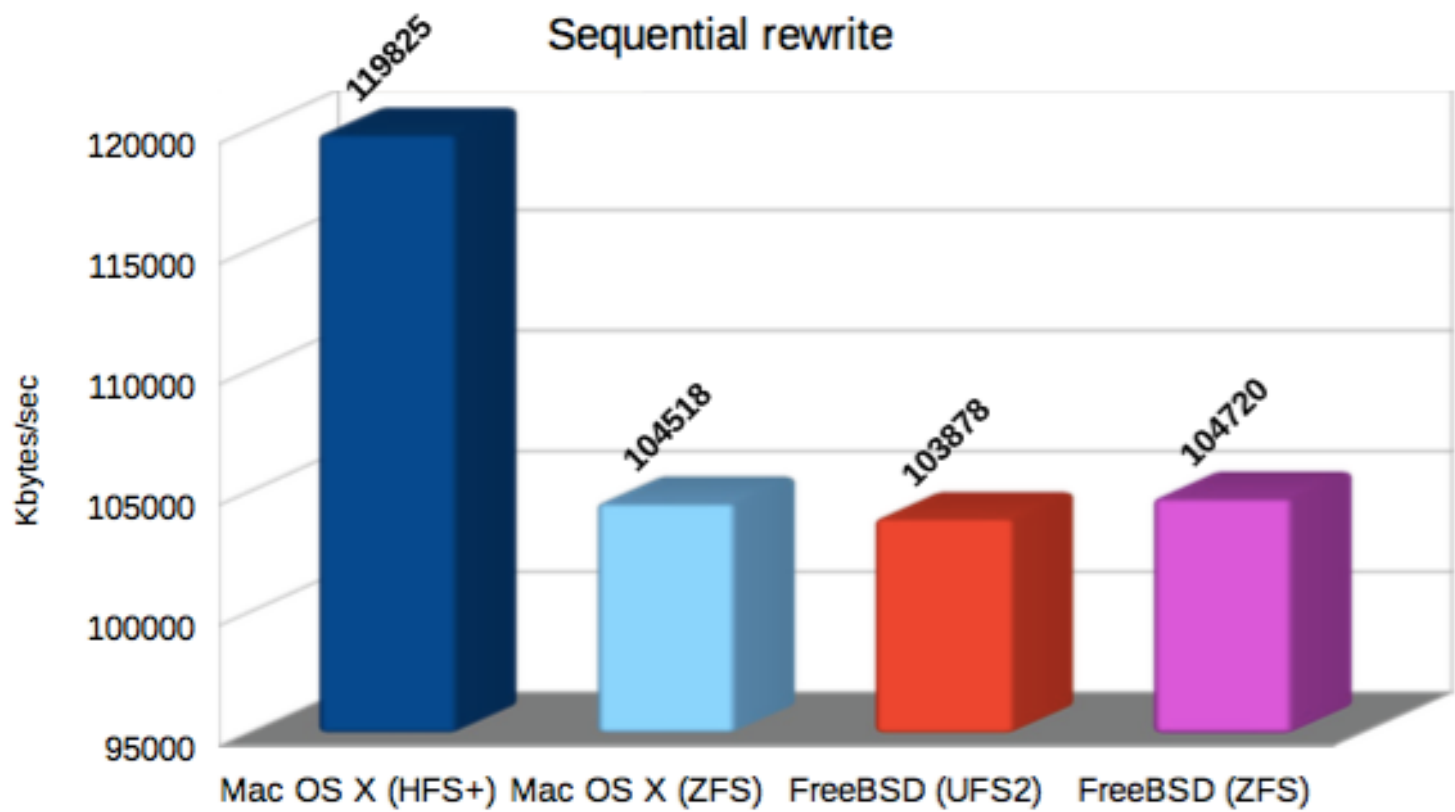


Figure 2. Sequential rewrite.

Fight Club

Mac OS X takes the advantage here, being faster than FreeBSD in their native filesystems. On ZFS, they have practically the same speed.

Sequential read

In this test, Bonnie++ sequentially reads the data it previously wrote.

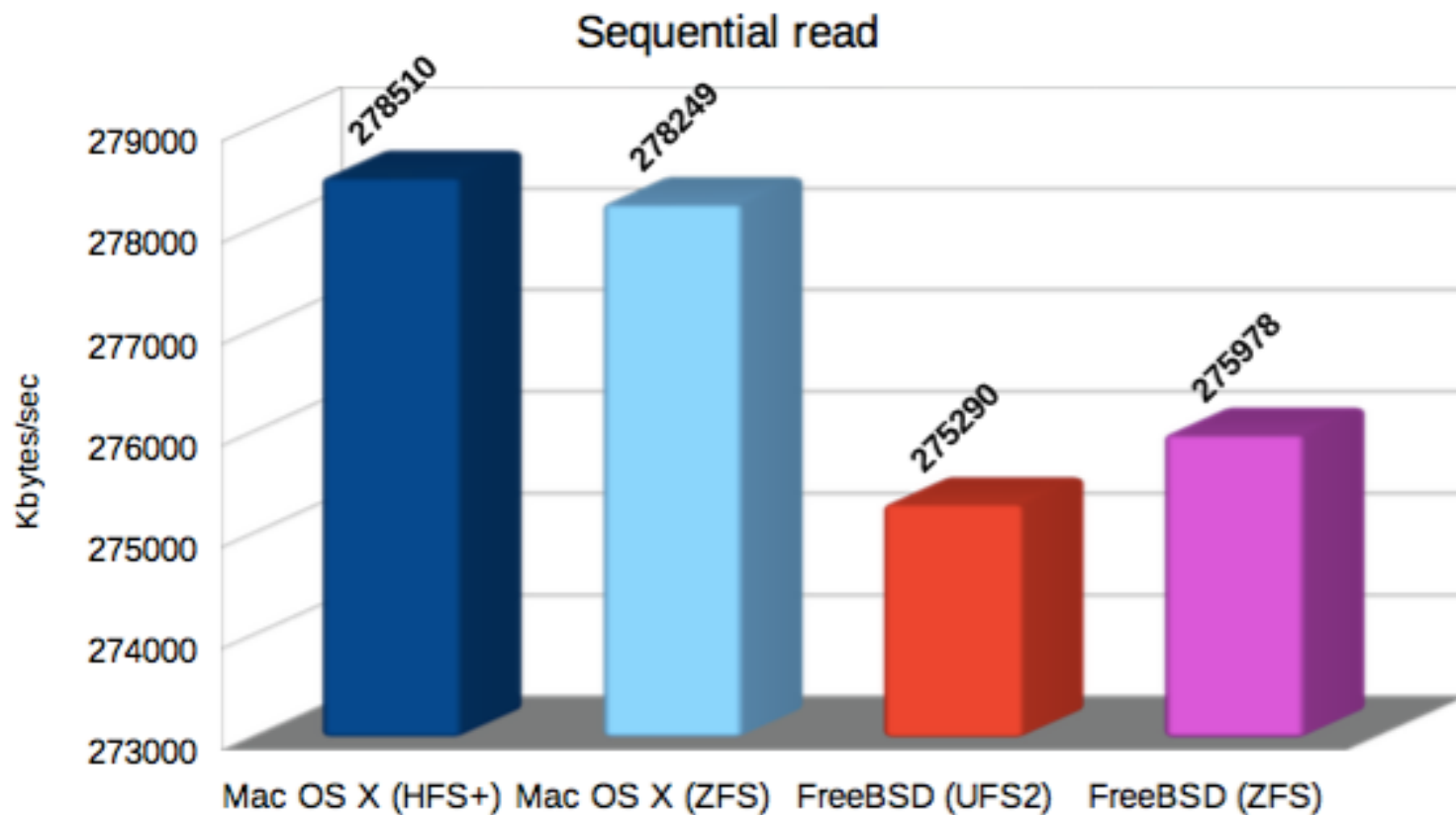


Figure 3. Sequential read.

Mac OS X shows to be a little faster (less than 2%) than FreeBSD.

Random seeks

In this test, Bonnie++ takes the previously created files and checks how many times it can randomly seek inside them.

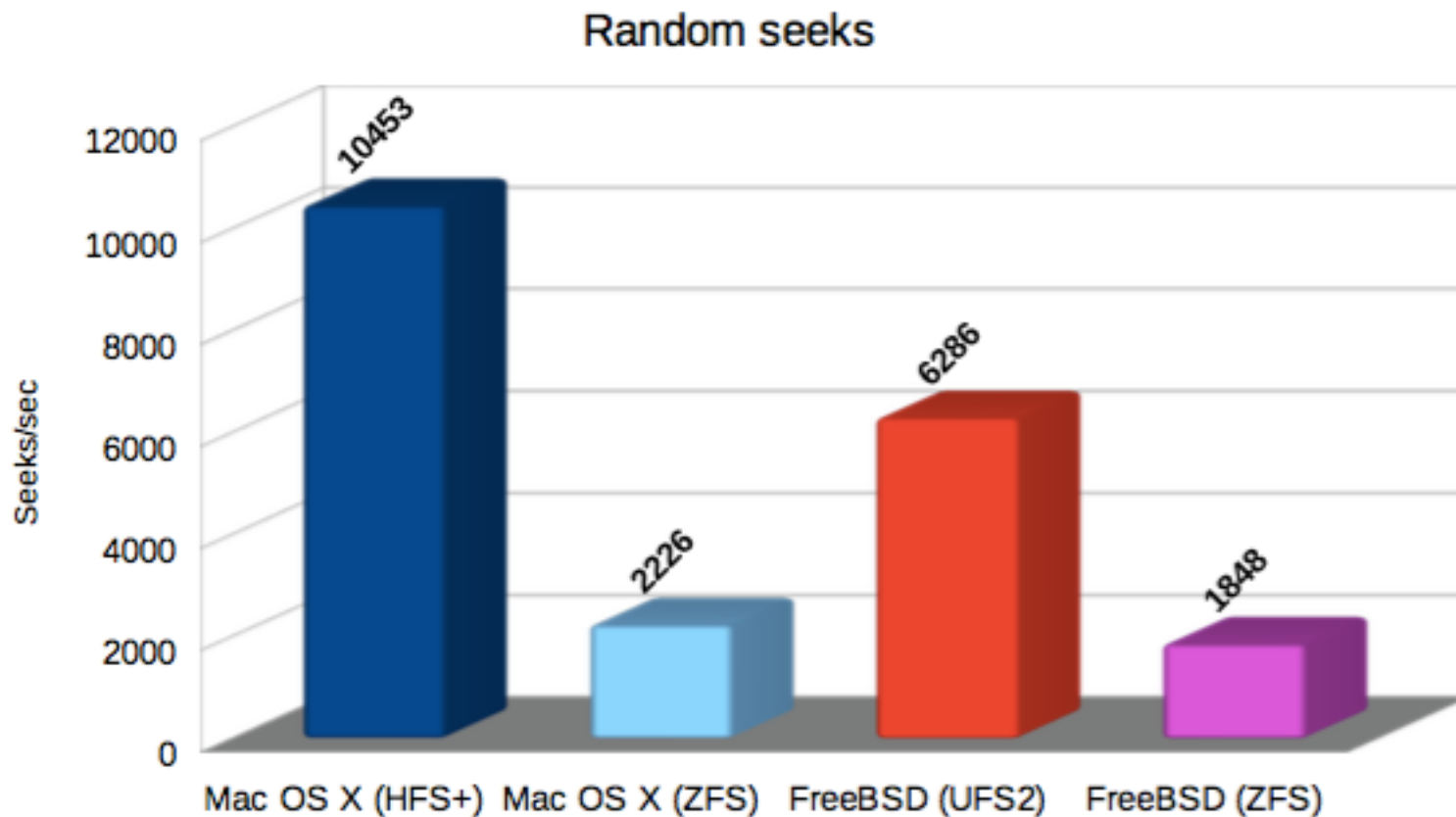


Figure 4. Random seeks.

Native filesystems are quite a bit better in this than ZFS, and Mac OS X gets a huge advantage over FreeBSD. This is probably because of filesystem code optimization and caching of data structures, as well as the native filesystems being less prone to fragmentation after the rewrite than ZFS.

Throughput conclusion

If your usage case would be to create and serve big files in small numbers, like as a remote media server, Mac OS X will give a bigger overall throughput, and if you don't require the features offered by ZFS, HFS+ gives an even better one.

Sequential file creation

In this test, Bonnie++ creates several files and subdirectories sequentially.

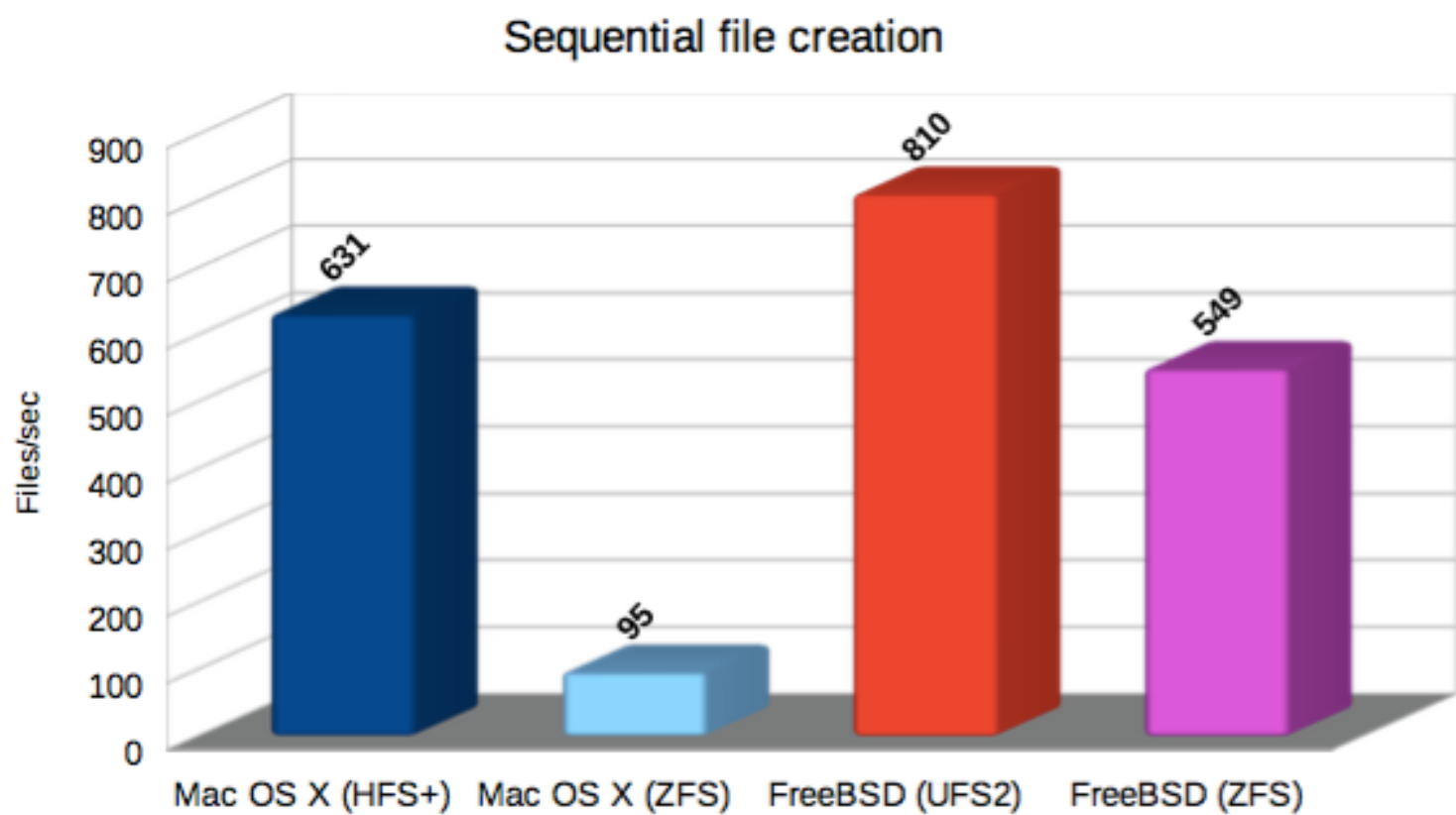


Figure 5. Sequential file creation.

Here, FreeBSD takes the advantage, even more markedly if using ZFS.

Sequential directory read

In this test, Bonnie++ traverses the previously created files and subdirectories.

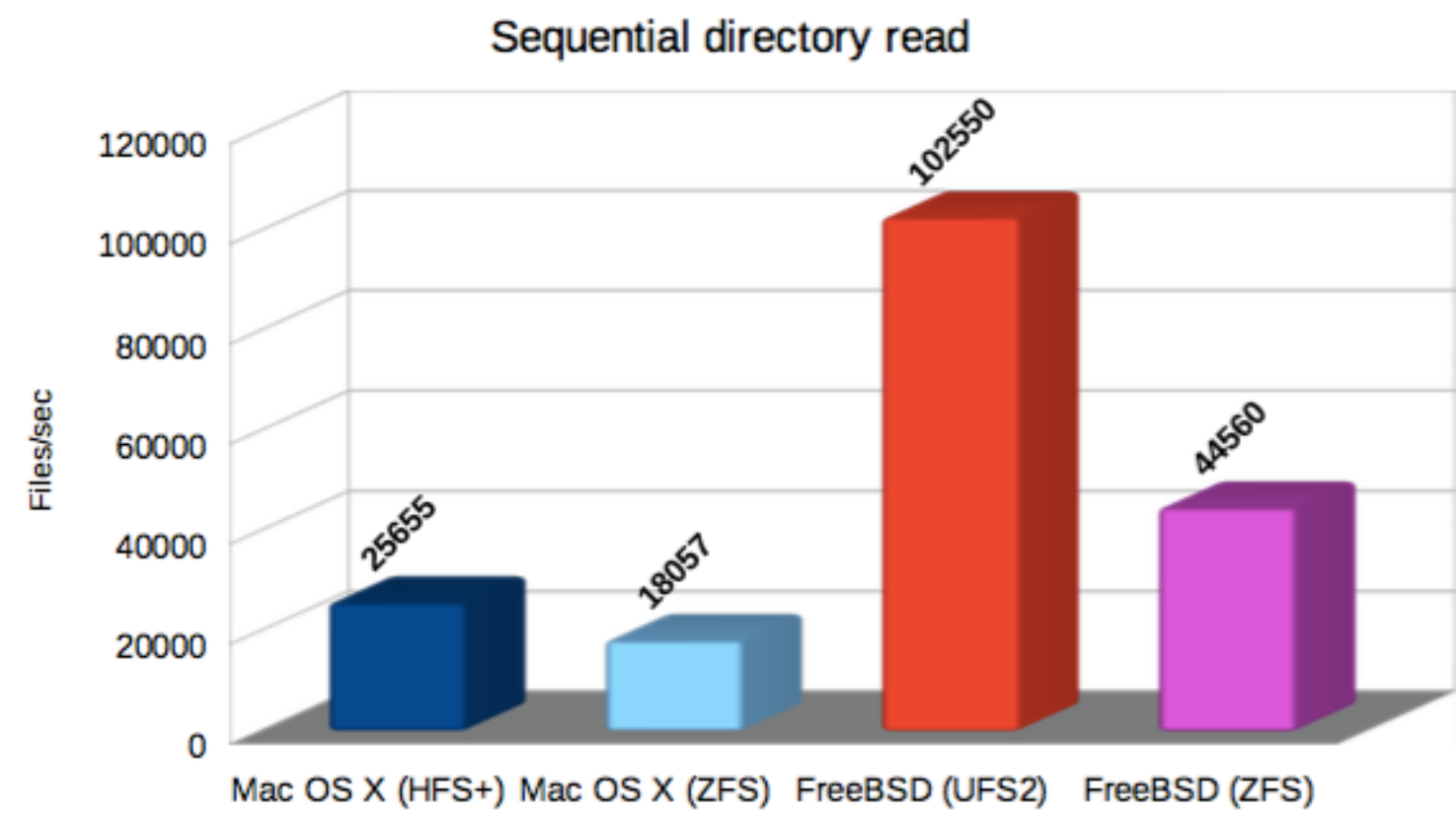


Figure 6. Sequential directory read.

FreeBSD shows an even better margin than the previous test.

Sequential file deletion

In this test, Bonnie++ removes the previously created files and subdirectories.

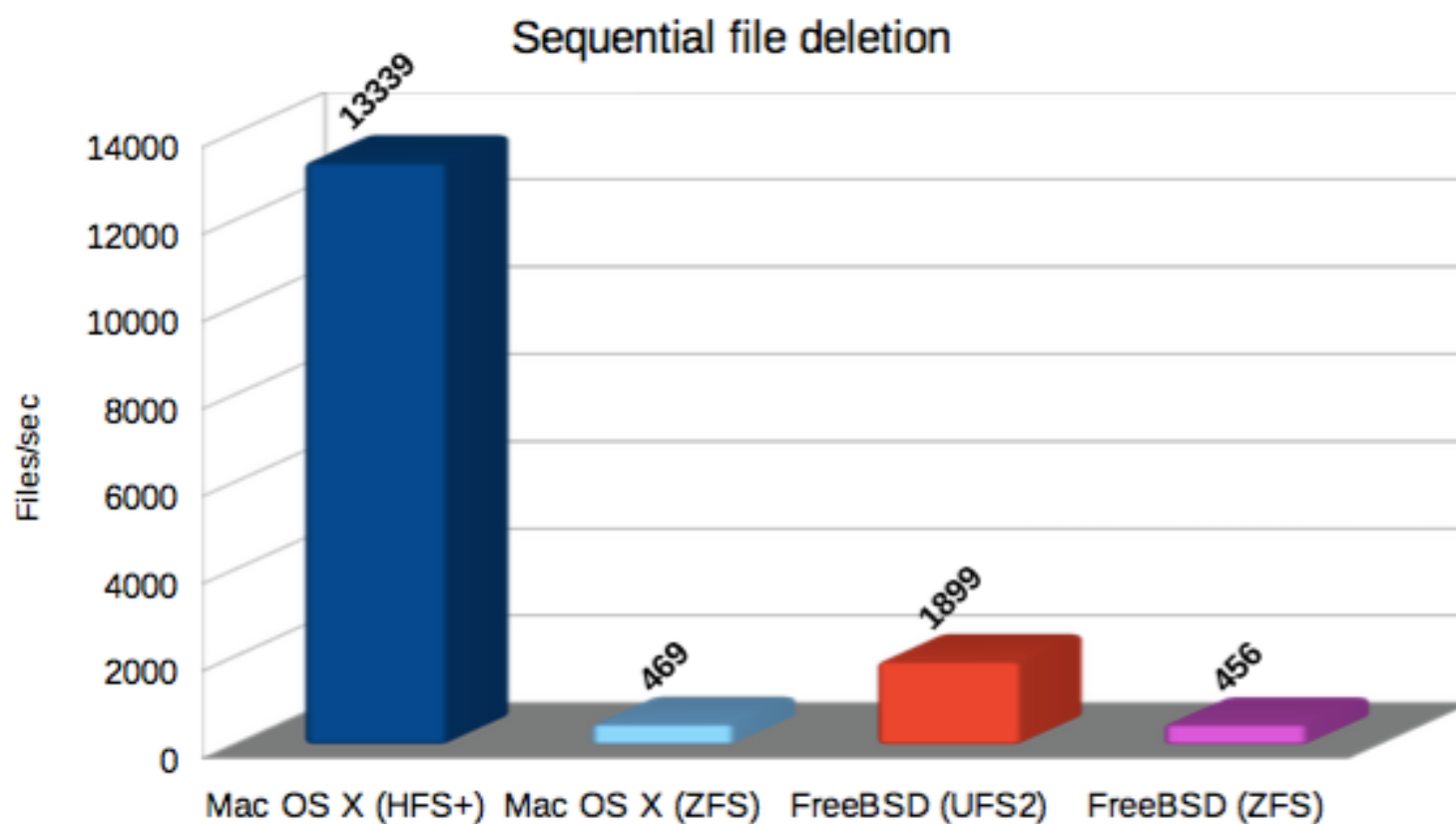


Figure 7. Sequential file deletion.

Here, HFS+ shows the advantage of using a single B-Tree for everything, where deletes become easier as there are less nodes to take account for, while ZFS shows the disadvantage of copy-on-write, as it has to create a new copy of the blocks containing the directories on each deletion.

Random file creation

In this test, Bonnie++ creates new files and subdirectories randomly choosing their location.

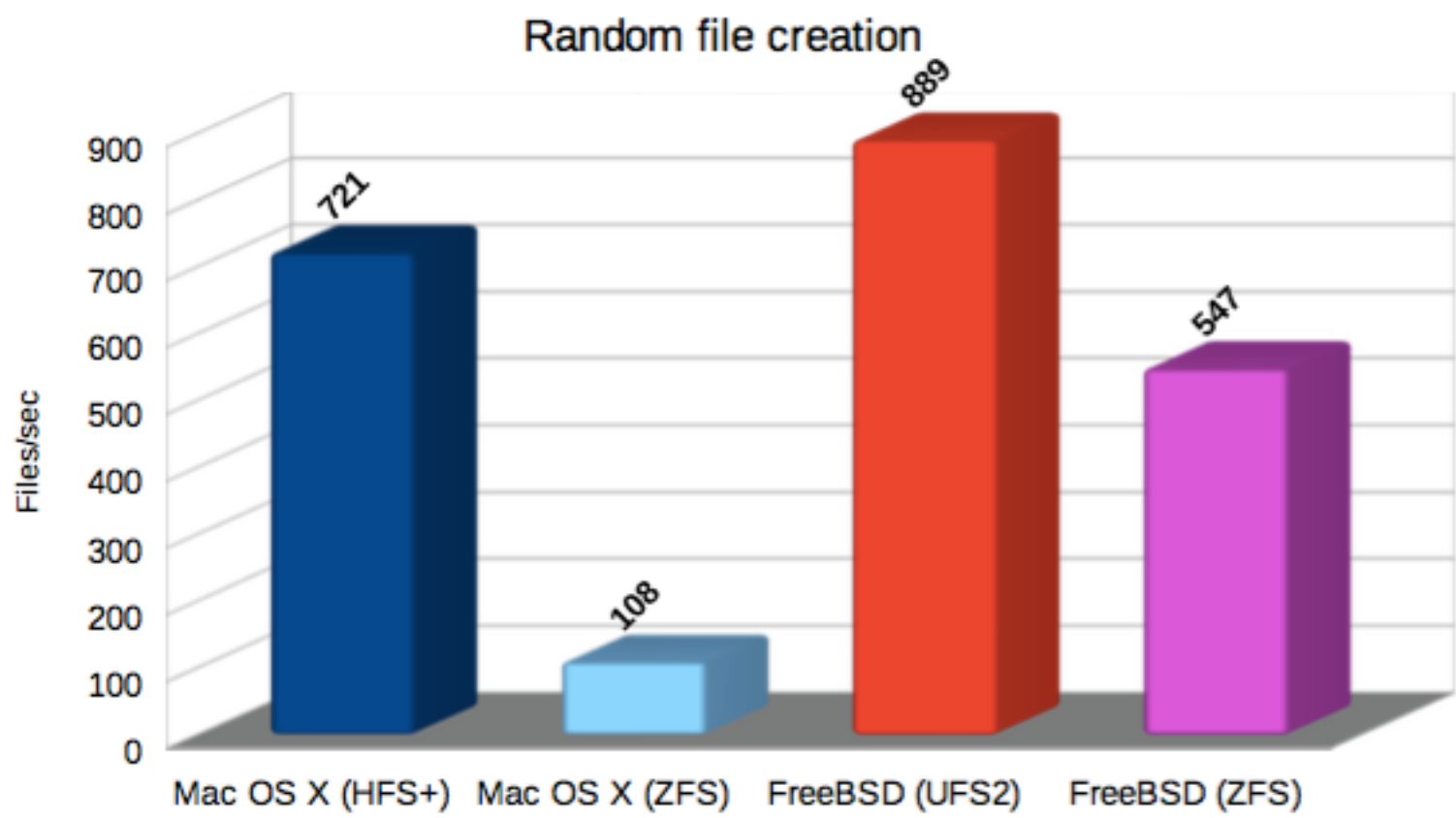


Figure 8. Random file creation.

In this test, contrary to the sequential equivalent, FreeBSD takes the lead, getting significantly faster in ZFS over Mac OS X and slightly faster in UFS vs HFS+.

Random directory read

In this test, Bonnie++ randomly accesses the previously created files and subdirectories.

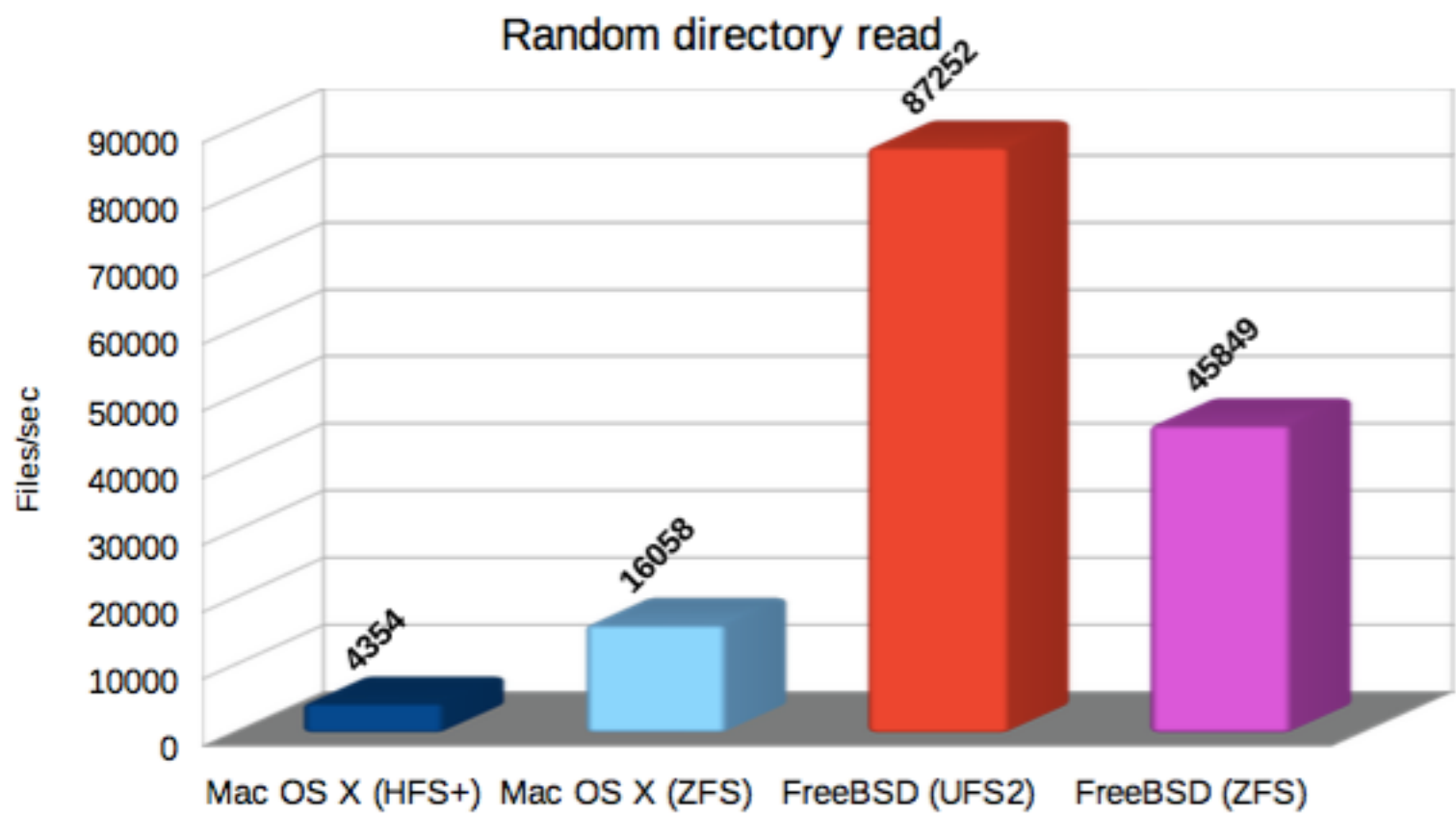


Figure 9. Random directory read.

Fight Club

In this test, HFS+ shows its inherent problem of having a single B-Tree that must be wholly traversed always, while ZFS and UFS, using a simple directory structure, are faster. Also, FreeBSD takes the lead being significantly faster than Mac OS X.

Random file deletion

In this test, Bonnie++ randomly deletes the previously created files and subdirectories.

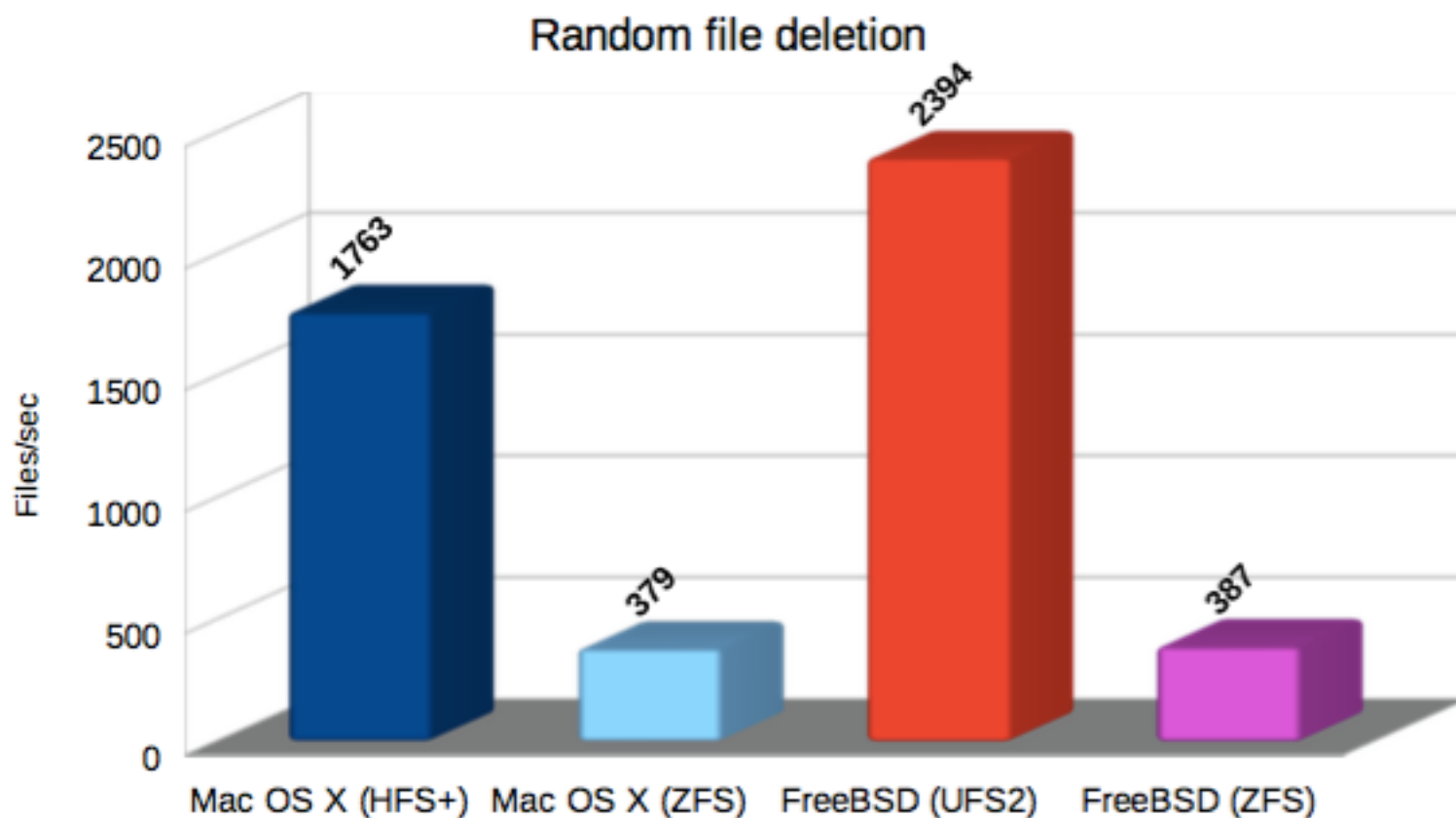


Figure 10. Random file deletion.

Once again, ZFS shows the disadvantages of copy-on-write, and FreeBSD takes the lead in speed with a significant, but not huge, difference.

Big number of files - conclusion

If your usage case would be to create and serve a lot of small files, like for example a mail or web server, FreeBSD has a significantly big advantage over Mac OS X, and you can discard UFS as in most cases ZFS is not so slow as to be in a disadvantage over UFS.

The next tests were done using Phoronix Test Suite. I will talk only about the tests that run correctly on both operating systems. You can see the whole results in <https://openbenchmarking.org/result/1605120-GA-0139603641>.

SQLite

This test measures the time to perform a pre-defined number of operations on an indexed SQLite database.

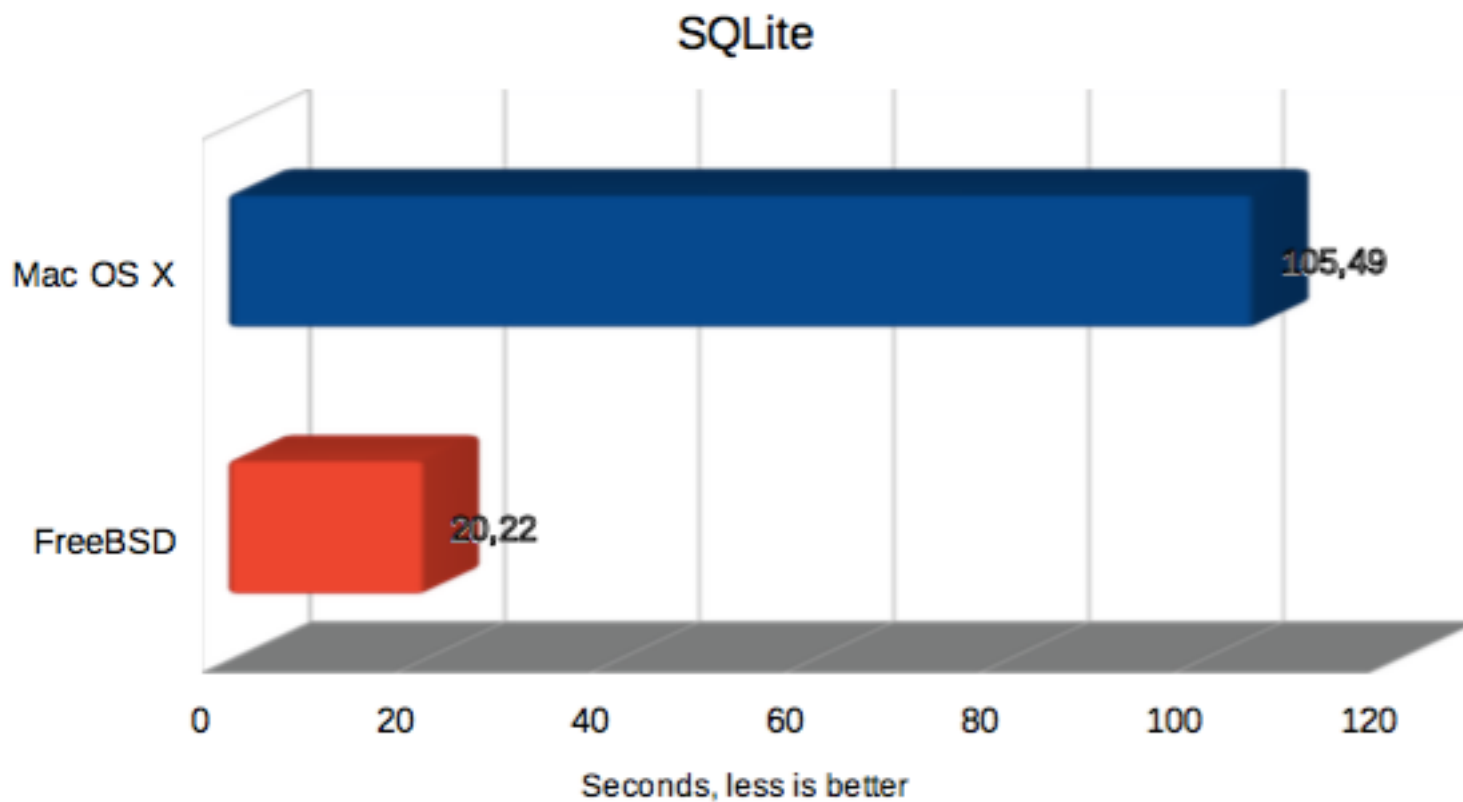


Figure 11. SQLite database.

FreeBSD goes five times faster than Mac OS X.

C-Ray

This test benchmarks pure floating point performance using a multi-threaded ray tracing to generate a 1600 x 1200 pixels image.

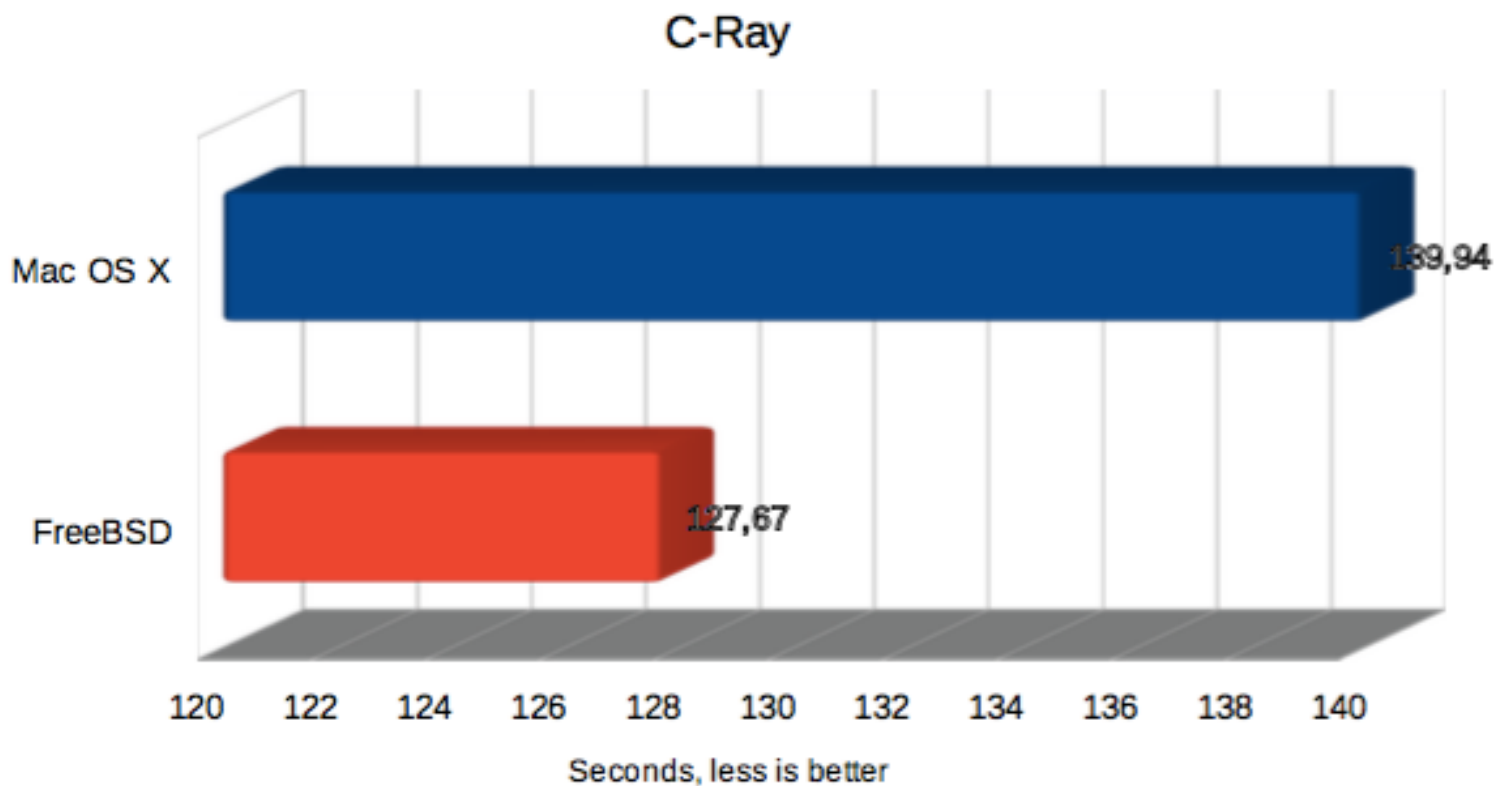


Figure 11. C-Ray benchmark.

Here Mac OS X is 10% slower than FreeBSD.

Fhourstones

This integer benchmark solves positions in the game of Connect-4, as played on a vertical 7x6 board. By default, it uses a 64Mb transposition table with the two big replacement strategy. Positions are represented as 64-bit bitboards, and the hash function is computed using a single 64-bit modulo operation, giving 64-bit machines a slight edge. The alpha-beta searcher sorts moves dynamically based on the history heuristic.

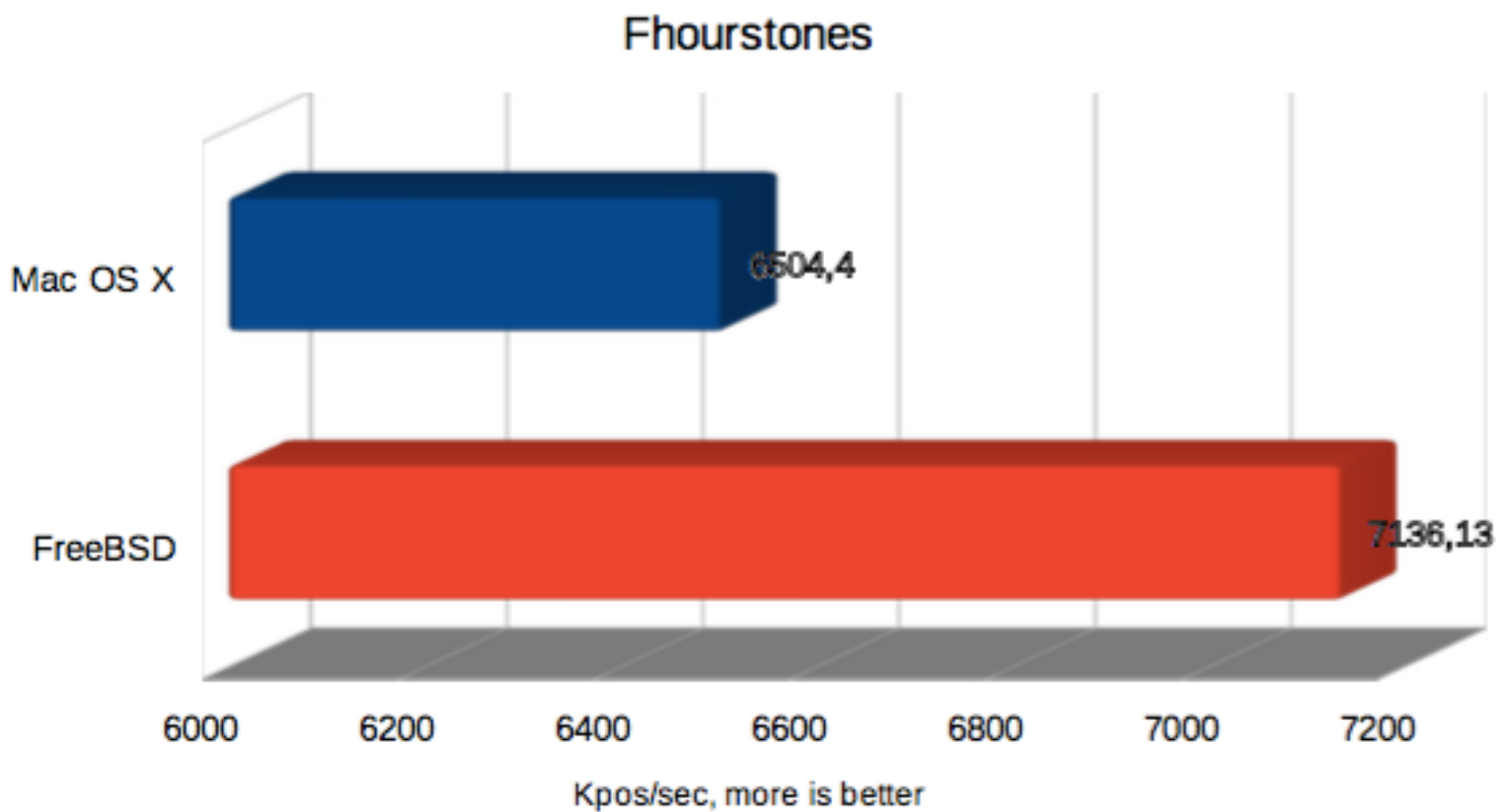


Figure 12. Fhourstones benchmark.

Once again FreeBSD gets 10% faster than Mac OS X.

Gzip

This test measures the time needed to compress a file using gzip.

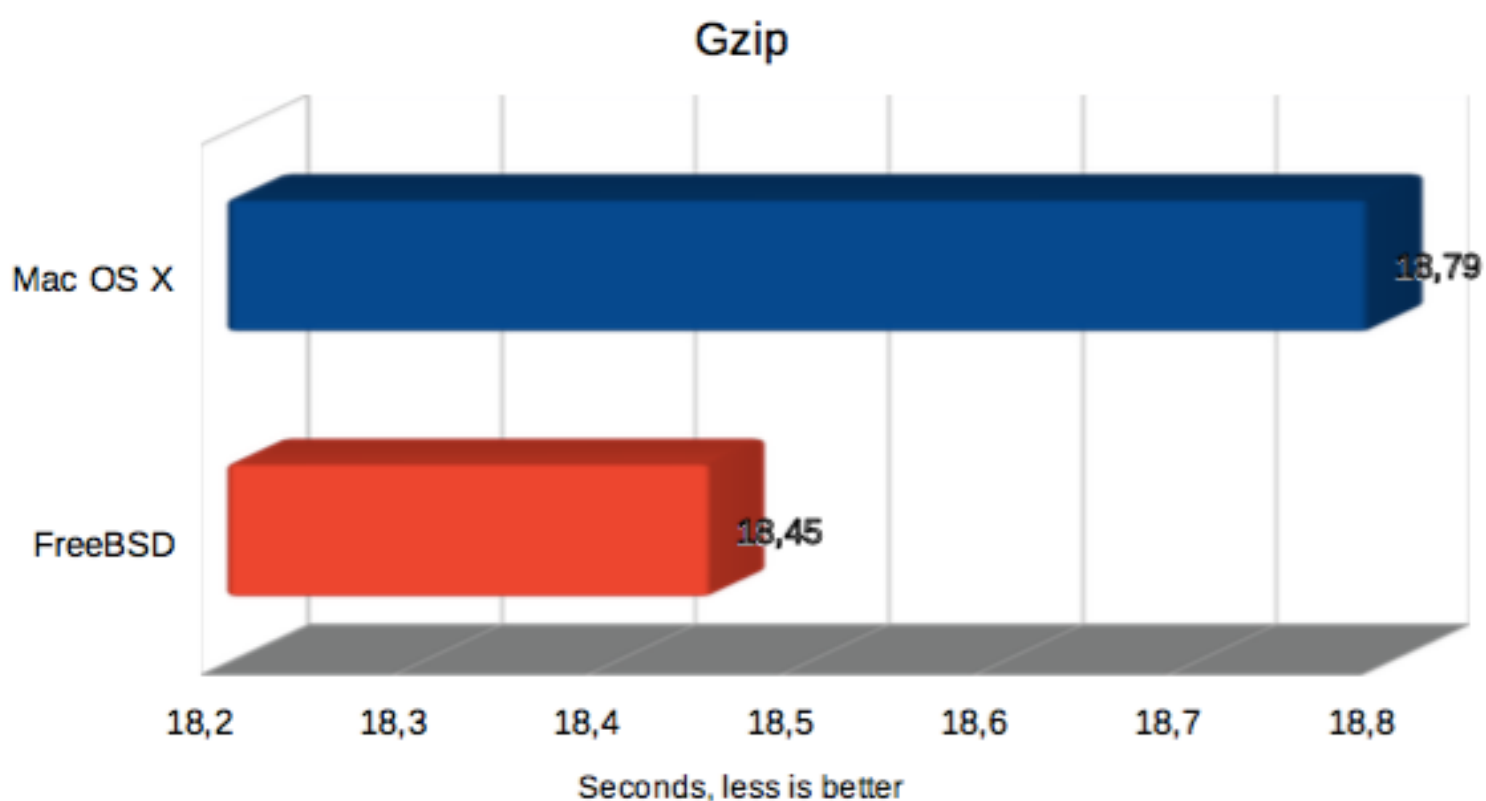


Figure 13. gzip compression.

OpenSSL

This test measures the 4096-bit performance of OpenSSL using the RSA algorithm.

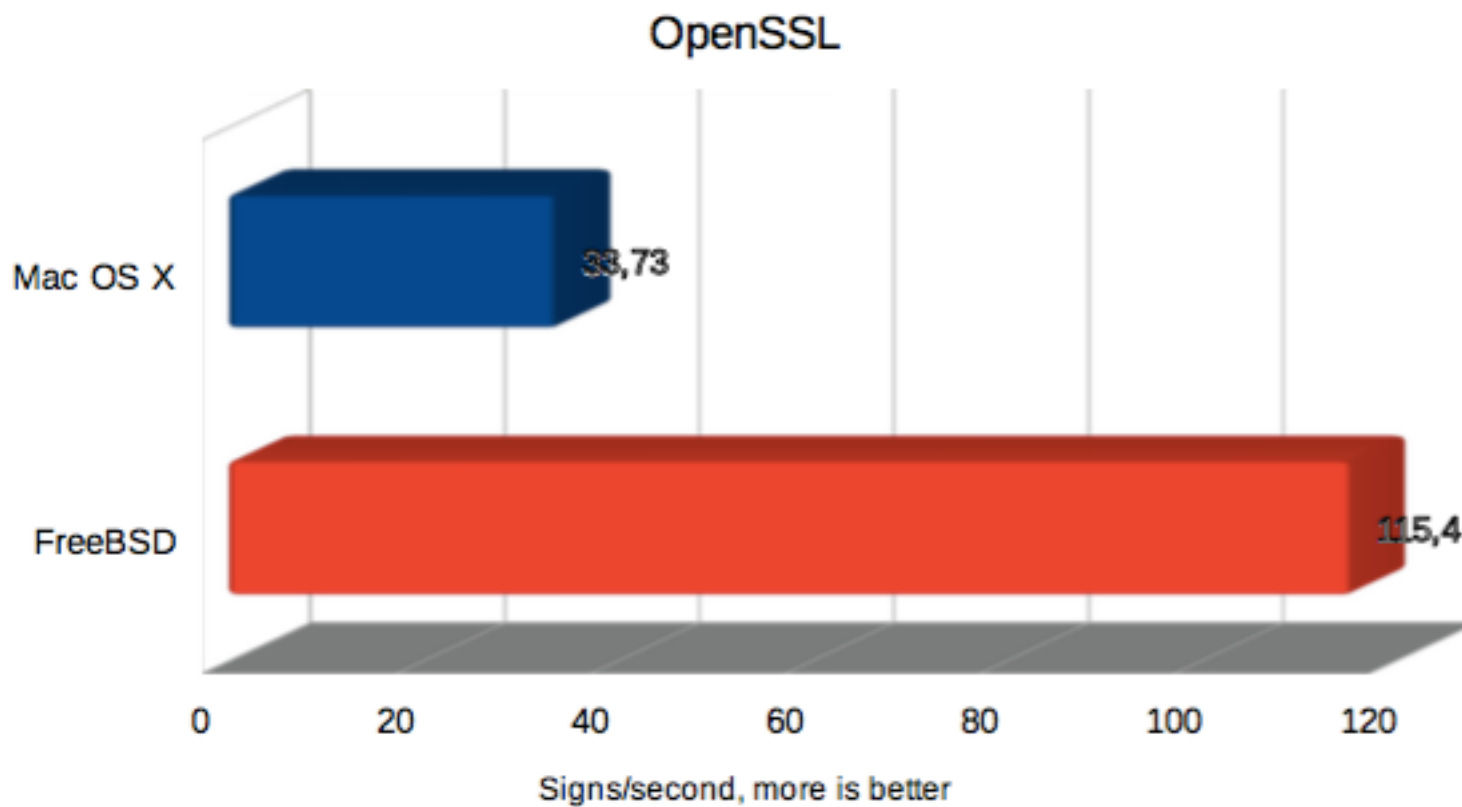


Figure 14. 4096-bit performance of OpenSSL using the RSA algorithm.

FreeBSD is able to do the test three times faster than Mac OS X.

Processing power - conclusion

On the side of raw processing power and performance, FreeBSD shows a clear advantage over Mac OS X, from 10% to 500% faster.

Overall conclusion

All benchmark scenarios put FreeBSD in the lead, except on the sequential file tests, with a huge difference. So, if you do not need something that exists only on Mac OS X, it seems that FreeBSD will get more juice out of your old Macintosh hardware.



About the Author:

Born in Canary Islands, transgender girl, independent and open source developer, computer historian, emulator lover, file system guru, Apple Certified Macintosh Technician, and .NET fan.

Server Automation for NodeJS over SSH with Nodemiral

by Ray Mahangoe

Nodemiral is like Ansible, an automation tool based on Javascript/NodeJS. While Ansible has been around for a while, Nodemiral is still young (though Javascript has been in the loop for a while). Why do I use Nodemiral? Well, I've been using Ansible for a while and I like the way Ansible works, like access with ssh, playbooks and run script on a server, etc., and there is no need to install a client on every server.

The point is, I like NodeJS and where ever it's possible, I make scripts to do some automation in NodeJS and when I saw this package “Nodemiral” for NodeJS, I started playing with it.

In my opinion, Nodemiral is still a work in progress and not completely ready for production uses, like Ansible or Puppet are, although it's good to play and experiment with. For some simple automation stuff for small/medium organizations that don't have a complicated infrastructure, Nodemiral can do a good job.

Nodemiral is nice but I will not compare it yet with Ansible, Puppet or other automation tool for now.

If you still want to try and use it, test it pretty good before automating your server task with Nodemiral, in this case go ahead and give Nodemiral a chance.

In my case, I'm running a Nodemiral test environment with 10 VMs, mixed with Linux (Debian, CentoS) and FreeBSD 10 & 11. It works fast and I feel at home with Javascript/Nodejs. ;-).

I think there is a good chance that this automation tool based on NodeJS will get up against the bigger tools in the future. Till that time I'll keep playing and testing automation tools based on NodeJS.

Features:

- Support connecting to any Unix remote server
- Authenticate with password (using sshpass) or with a pem file
- Can work with multiple servers at once
- Supports, copy, execute and executeScript at core methods Familiar NodeJS API

Requirements and Installation:

Requirements:

- Some basic knowledge with Javascript/NodeJS.
- Nodemiral is a package for NodeJS and the requirements are the same as for Javascript.
- You need to installed nodejs on your main server/workstation from where you want to control/ manage all your servers. There are plenty of tutorials on how to install Nodejs on the internet for Unix / Linux / OSX and Windows.

Installation:

```
nodemiral:  
  
$npm install  
  
nodemiral
```

Create a script for example nodemiral.js and past the code into it and save it.

This script will: login to a linux/unix server 'hostname' and will execute the command "uname -a"

Example:

```
var nodemiral = require('nodemiral');  
  
var session = nodemiral.session('hostname', {username: 'root', password: 'password'});
```

```
session.execute('uname -a', function(err, code, logs) {  
  
  console.log(logs.stdout);  
  
});
```

Explanation:

```
hostname = hostname or ip address  
  
auth = object containing following fields: `username` and (`password`  
or `pem`) err = err if exists  
  
code = status code of the ssh process  
  
logs = {stdout: 'stdout logs', stderr: 'stderr logs'}
```

Run the script called nodemiral.js

```
$node nodemiral.js
```

Other examples how to use Nodemiral

Session: Create a session to a remote server. You can invoke the following methods after you created a session example:

```
var session = nodemiral.session(hostname, auth, options);  
  
hostname = hostname or ip address  
  
auth = object containing following fields: `username` and (`password`
```



```
or `pem`) options = object of options described below.
```

```
-   ejs = ejs options with ejs fields

-   ssh = object whose key and value will be passed as -o key:
value to any ssh session. For example: { 'StrictHostKeyChecking':
'no', 'UserKnownHostsFile': '/dev/null' }
```

Session.execute: execute given shell command on the remote server:

```
var session = session.execute(shellCommand, options, callback);
```

```
shellCommand = shellCommand options = {onStdout, onStderr}
```

```
callback = callback containing following parameter
```

Session.executescript: execute a local shell script in the remote server. You can template shell script with <https://github.com/tj/ejs>

```
var session = session.executeScript(localScriptFile, options, call-
back);
```

```
localScriptFile = localScriptFile
```

```
options.vars = variables to the template if uses ejs in the script
callback = callback containing following parameters
```

Session-copy: copy a file from local machine to the remote machine. Supports binary files too. Support EJS templating with non-binary files.

```
var session = session.copy(localFile, remoteFileLocation, options,
callback)

localFile = localFile

remoteFileLocation = remoteFileLocation options.vars = templateVars
options.progressBar = show progress bar

callback = callback containing following parameters
```



About the Author:

My name is Ray I live in Amsterdam the Netherlands.
I started my IT future back in 1992 with playing around with a Atari 1024ST and MSDOS emulator.

Its was so fascination for me that I started to do more research and

selfstudy en got Novell NetWare administrator Certification in 1994.

In 2012 I started to play with DragonFly, OpenBSD and FreeBSD. I felt directly home with FreeBSD and I started working for the European Aerospace in The Netherlands and they uses FreeBSD as main server

for Storage, Web, PHP, Mysql, Ldap authentication, Firewall etc and CentOS for Matlab cluster.

In 2014 I went to an other company UPC “bought by AMC Network” and I implemented FreeBSD & ZFS and GlusterFS as one big storage pool of 18PB of capacity.

These days my main focus and interest is, FreeBSD, ZFS, GlusterFS, NodeJS and Openstack.

As hobby I do make Electronical Music and I also give Dance Workshop and Contact Improvisation trainings. ;-)



FREENAS MINI STORAGE APPLIANCE

IT SAVES YOUR LIFE.



HOW IMPORTANT IS YOUR DATA?

Years of family photos. Your entire music and movie collection. Office documents you've put hours of work into. Backups for every computer you own. We ask again, *how important is your data?*

NOW IMAGINE LOSING IT ALL

Losing one bit - that's all it takes. One single bit, and your file is gone.

The worst part? **You won't know until you absolutely need that file again.**



Example of one-bit corruption

THE SOLUTION

The FreeNAS Mini has emerged as the clear choice to save your digital life. **No other NAS in its class offers ECC (error correcting code) memory and ZFS bitrot protection to ensure data always reaches disk without corruption and *never degrades over time.***

No other NAS combines the inherent data integrity and security of the ZFS filesystem with fast on-disk encryption. No other NAS provides comparable power and flexibility. The FreeNAS Mini is, hands-down, the best home and small office storage appliance you can buy on the market. **When it comes to saving your important data, there simply is no other solution.**

The Mini boasts these state-of-the-art features:

- 8-core 2.4GHz Intel® Atom™ processor
- Up to 16TB of storage capacity
- 16GB of ECC memory (with the option to upgrade to 32GB)
- 2 x 1 Gigabit network controllers
- Remote management port (IPMI)
- Tool-less design; hot swappable drive trays
- FreeNAS installed and configured



<http://www.iXsystems.com/mini>



FREENAS CERTIFIED STORAGE



With over six million downloads, FreeNAS is undisputedly *the* most popular storage operating system in the world.

Sure, you could build your own FreeNAS system: research every hardware option, order all the parts, wait for everything to ship and arrive, vent at customer service because it *hasn't*, and finally build it yourself while hoping everything fits - only to install the software and discover that the system you spent *days* agonizing over **isn't even compatible**. Or...

MAKE IT EASY ON YOURSELF

As the sponsors and lead developers of the FreeNAS project, iXsystems has combined over 20 years of hardware experience with our FreeNAS expertise to bring you FreeNAS Certified Storage. **We make it easy to enjoy all the benefits of FreeNAS without the headache of building, setting up, configuring, and supporting it yourself.** As one of the leaders in the storage industry, you know that you're getting the best combination of hardware designed for optimal performance with FreeNAS.

Every FreeNAS server we ship is...

- » Custom built and optimized for your use case
- » Installed, configured, tested, and guaranteed to work out of the box
- » Supported by the Silicon Valley team that designed and built it
- » Backed by a 3 years parts and labor limited warranty

As one of the leaders in the storage industry, you know that you're getting the best combination of hardware designed for optimal performance with FreeNAS. **Contact us today for a FREE Risk Elimination Consultation with one of our FreeNAS experts.** Remember, every purchase directly supports the FreeNAS project so we can continue adding features and improvements to the software for years to come. **And really - why would you buy a FreeNAS server from *anyone* else?**



FreeNAS 1U

- Intel® Xeon® Processor E3-1200v2 Family
- Up to 16TB of storage capacity
- 16GB ECC memory (upgradable to 32GB)
- 2 x 10/100/1000 Gigabit Ethernet controllers
- Redundant power supply

FreeNAS 2U

- 2x Intel® Xeon® Processors E5-2600v2 Family
- Up to 48TB of storage capacity
- 32GB ECC memory (upgradable to 128GB)
- 4 x 1GbE Network interface (Onboard) - (Upgradable to 2 x 10 Gigabit Interface)
- Redundant Power Supply

<http://www.iXsystems.com/storage/freenas-certified-storage/>



GhostBSD - Easy to Use, Yet Powerful

by Kalin Staykov

When I first heard about GhostBSD, I thought it may involve a shady distribution that is all about security. Okay, in fact, at first I thought about actual ghosts, but let's not dive into that. The name comes from "Gnome hack operating system technology BSD". This project is all about putting a nice desktop environment with all the security perks of having a BSD system under the hood.

Later on, I'll tell you how you can install it and play with it a bit, but let's first see why would one want to bother with it. If you are familiar with distributions from the Linux world, you might know Linux Mint. Its goal is pretty close to what the creators of GhostBSD were aiming for.

It's simple, it's pretty and it works

Now let's talk about Mate – one of the desktop user interfaces you can pick from.

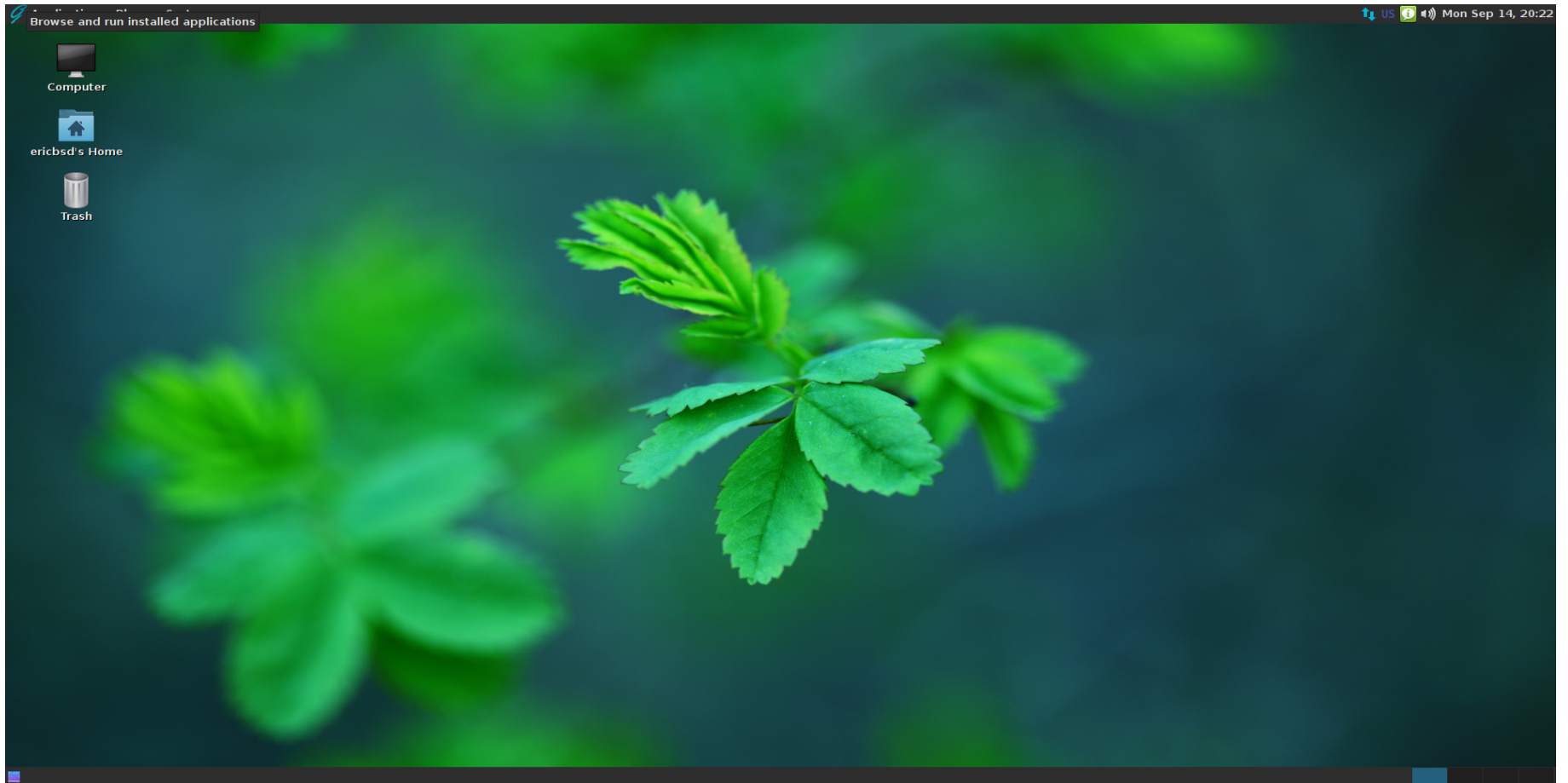


Figure 1. GhostBSD Desktop.

If you're a tea lover like me, you'll be delighted to learn about where its name comes from. It comes from yerba maté (pronounced Ma-Tay), which is widely known as a source of a delightful beverage. The origin of Mate's interface goes back in time to earlier development stages of the GNOME desktop, or to be exact – GNOME 2. Reborn and rejuvenated, this desktop was developed by a community of more than 20 people.

Installing GhostBSD with Mate desktop

I'm not ashamed by the fact that the first time I installed FreeBSD, I wiped my entire hard drive. It was long ago with a very early version, but hey, the world of open software was tough back then! This doesn't sound like a good excuse and the good news is that you won't need any such excuses, because when you boot the installation media, you'll be greeted with a graphical installation screen. It will make the installation a piece of cake, but be mindful – every dungeon has its goblins.



Figure 2. GhostBSD Installation.

Okay, no need to pick an axe before hitting enter – just do it and think happy thoughts, while some weird text shows up. It won't ask for anything, so just wait it out.



Figure 3. GhostBSD Installation.

Welcome to Mate

Yes, it's that easy – now you can play with the distribution and see if you actually like it. Keep in mind that this is Live boot and nothing is installed on your disk yet. Take your time to explore. That's my favorite part. Out of the box, you'll see the Firefox browser installed. You'll also have mail and messenger clients, so if you decide that some Internet experience is important to you, go on and try it.

Actual Installation

Once you're ready and you like what you see, you can proceed to the installation. Start up the installer, which is located on your desktop. You'll have to select several things during the installation. The first three are fairly simple:

- Your language
- Your keyboard layout
- Time zone

And now comes the tricky hard drive selection option. That's where you should be careful. It looks like that:

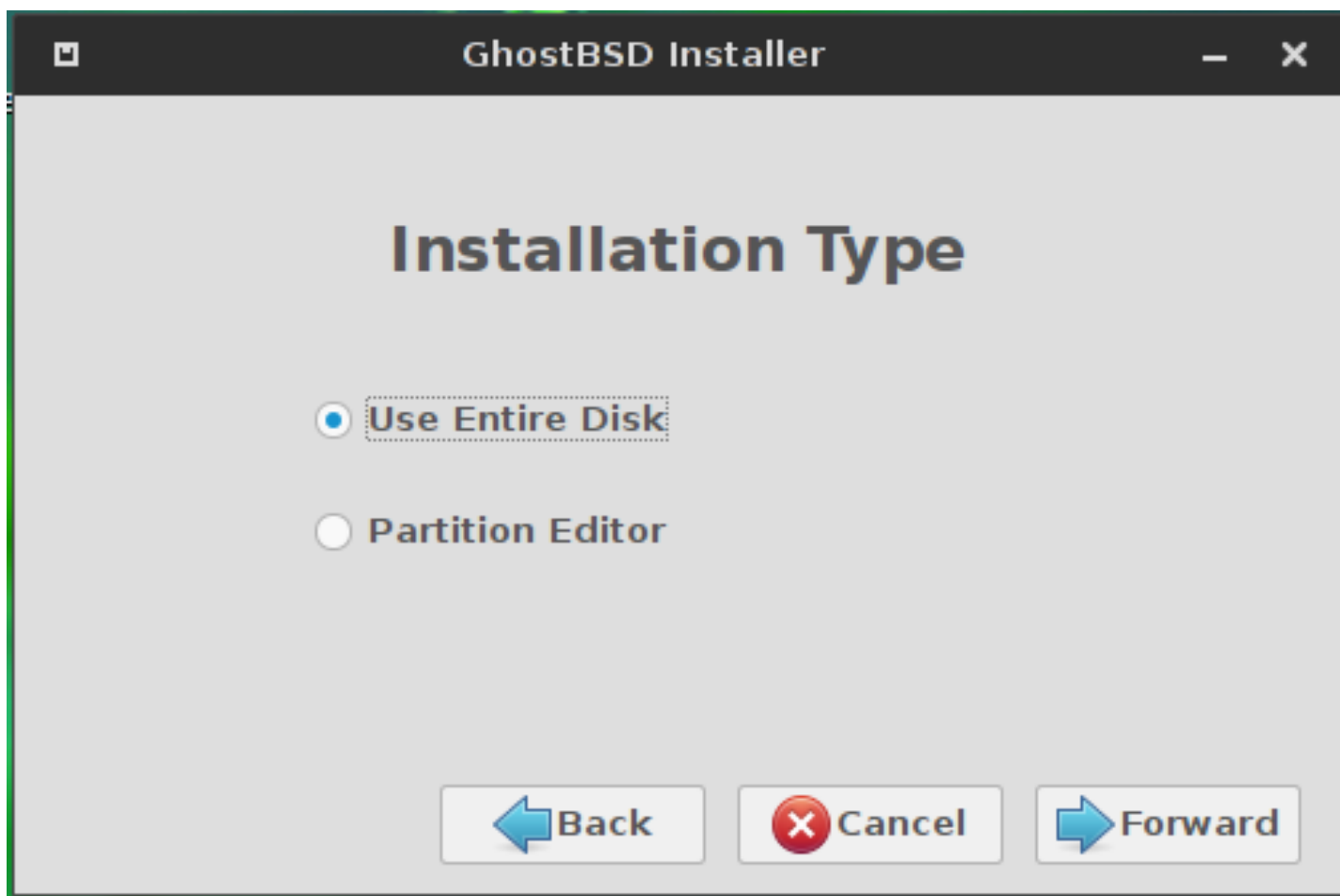


Figure 4. GhostBSD Installation Type.

You should be prepared for that, so I'll take a moment to explain. If you have a single disk, using the entire disk will mean that you don't need any data that's currently stored there. With this option, you also commit to use this distribution as your only option and not dual boot. The dual boot is the ability to have more than one OS installation on one or more disks of the same system. Be mindful of your choice since the next steps can erase data on the disks you have.

The partition editor can enable more flexibility, but you'll need to be armed with some experience using such managers. If not, have a look at the wiki page of Custom Disk Partitioning Installation or look for some video tutorials that will show you how it's done. I will assume that you have some level of experience or the willingness to research this topic. It's very important to not experiment with hard drives that have valuable information. If that's the case, I would suggest you consult someone or start the IRC client from your desktop and talk to the people in the channel that will come up.

Once you make up your mind about partitioning, move to the next section and enter your root password. That's also the place to pick your boot manager. This decision is usually paired to how you use your disk, so I'll skip this explanation with the tip to use same level of help as you did on the previous step. User creation follows next. I recommend to use this non-privileged user whenever possible. Going with just root is not secure, and it might have serious implications to how you use this distribution.

Security is up to you and creating a non-privileged user is the first step to having a secure desktop

If all goes well, you should be seeing the soothing installation screen reporting progress. Take a moment to relax, grab a beer, or just watch the screen, if you're interested in some of the options this distribution provides. That's all you need for a successful GhostBSD installation. Let's make a small summary of what was required during the installation:

- Language and keyboard
- Time zone
- Disk partitioning
- Users and passwords

I could argue that these are the minimum things the user should know in order to make a nice installation of any operating system.

Tools and software

Out of the box, you'll get the Libre Office suite installed and some really nice applications that will

lserve your basic needs. Reviewing pictures, browsing the web and making documents is now available for you to dive into.

The OctoPkg package manager

If you need additional software, you'll need one important tool called OctoPkg. That makes up for the complexity of software installation on BSD and provides a nice GUI interface that will let you install anything you like very easily.

To start using this software, I suggest you go in the File menu and pick "Sync database" or hit the corresponding button on the left of the main panel. This option gets the latest updates for software repositories. It will provide important information about what your system needs in terms of updates and what is important to grab as soon as possible.

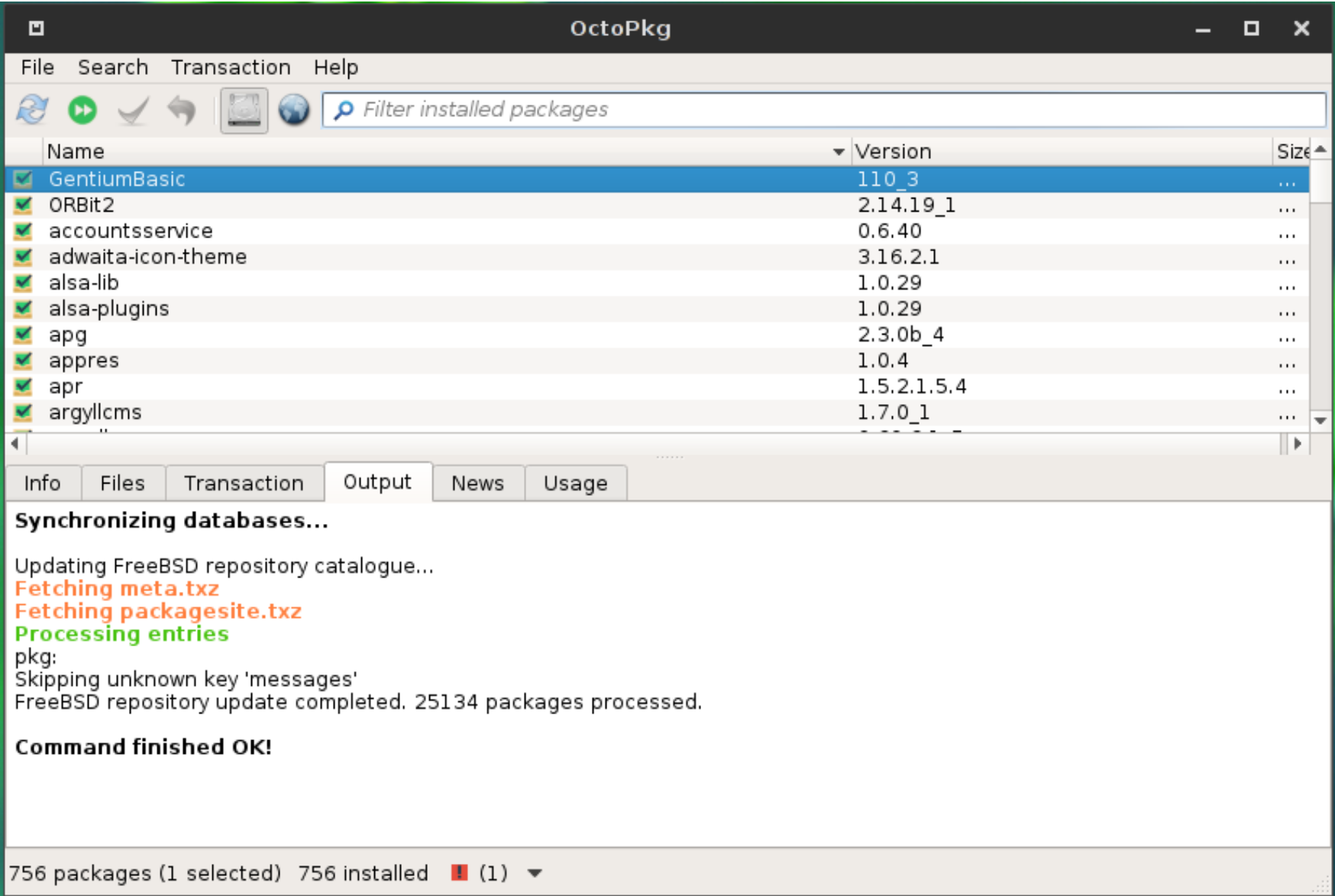


Figure 5. GhostBSD File Menu.

In my case, I have one update that shows red. I can click on the explanation mark and review the details. I strongly suggest you get the latest updates first, before installing new software. That will ensure that you have all dependencies setup correctly, the latest security patches installed and, in general, a good user experience with bug fixes and improvements.

Once you confirm the upgrade of the selected packages, the process is completely automatic.

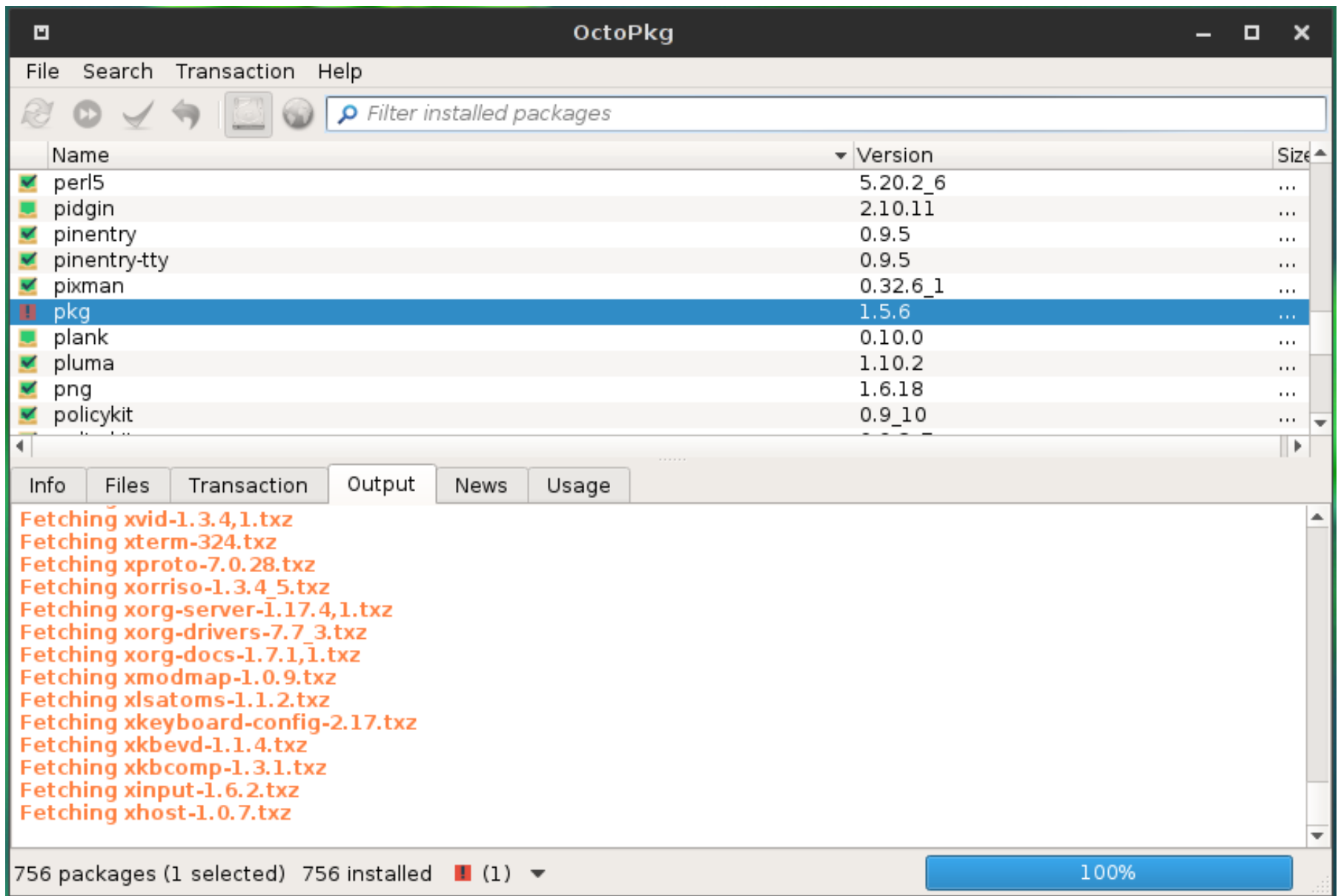


Figure 6. GhostBSD File Menu.

If you want to use BSD and you're looking for a good distribution that is easy to use – look no further. GhostBSD will give you an easy time dealing with one of the most sophisticated and advanced operating systems out there.

Keeping the system up to date

Although OctoPkg gives you a nice interface of getting your system up to speed with latest improvements and updates, I think it's nice to have a good awareness of other ways to do the job. Just like any other BSD, there is a tool called “freebsd-update” that can quickly get your system upgraded. There are two steps involved:

1. Fetching updates

```
freebsd-update fetch
```

2. Installing the updates

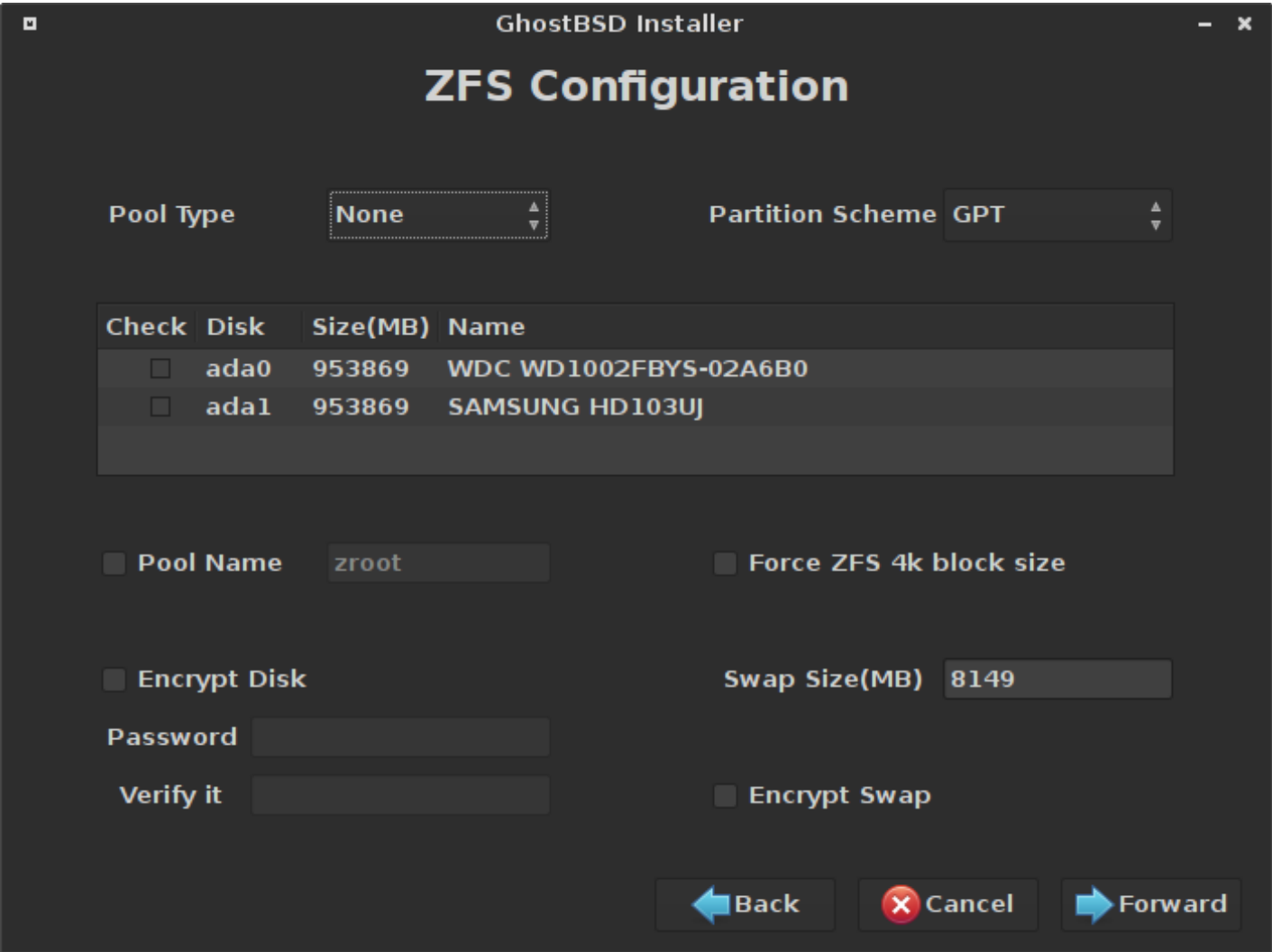
```
freebsd-update upgrade
```

Those steps are usually combined using directly:

```
freebsd-update fetch install
```

ZFS and getting a more resilient filesystem

The Zettabyte filesystem (ZFS) is one significant improvement that can provide an extra layer of resilience to your system. One key feature is that it is always consistent. That means it will never need to be checked for integrity and it will self-heal any damage it endures due to sudden power outages, silent data corruptions or bad write operations. It is much more than that – it can be encrypted, it has a full stack of RAID capabilities and it can be used as a pool for many mounts, which gives extra flexibility when you don't know which mount will grow.



That's all nice but GhostBSD doesn't provide support for root disk installation directly. At least not yet – it's planned for the new release but as of 10.1 it's not currently available. In the near future, when it comes in the stable release, users will be able to take advantage of it through the installation panel:

Figure 7. GhostBSD Installation - ZFS Configuration.

Software installation from source

There is one last thing I want to mention that I decided to leave for the end of this article.

It is optional for most users but if you need some special software that is not in the OctoPkg, there is still a chance for it being available in the BSD ports. To enable ports capability, you need to fetch and extract them using:

```
portsnap fetch  
portsnap extract
```

Those commands will populate the `/usr/ports` directory and provide an interface for building the package from source.

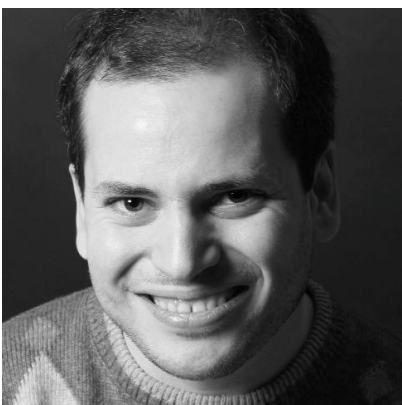
I am a master of making lots of typos when typing commands and I enjoy having a bit of fun noticing when I'm not hitting the keys in right order. One example is the "ls" command, which I often mistype with "sl" instead. Let's install Steam Locomotive from the ports gaming section. It will print a cute steam locomotive every time we type "sl". Let's install it using:

```
portinstall sl
```

Now try the "sl" command and enjoy the software you just installed from source. In case you are wondering, there are a few things that happen when you call `portinstall`:

1. The source code is downloaded from the repository. In the case of `sl`, the repository is hosted at Github.
2. The Make tool is engaged and based on the rules at `/usr/ports/<package>/Makefile` the build process compiles the source to binaries and distributes them (install rule) to your system.

Builds from source installations are a complex matter and users don't often dig into them. However, sometimes that's the only way of getting some software installed. It is likely that the software is both – available and latest if you install it that way. That's why having such knowledge is good, even for users who are not very experienced.



About the Author:

I'm very enthusiastic about all things open source. In my professional career I am a system administrator, developer, technical writer, DevOps. Nowadays, I prefer to enjoy some peace and quiet and spend my time with hobbies like photography and amateur radio.

Kalin Staykov, k.t.staykov@gmail.com

FreeNAS Getting Started Guide: Part 1, Planning and Installation

by Mark VonFange

This article is intended to serve as an introductory guide to assist FreeNAS users in planning, installation, configuration and administration for their FreeNAS storage systems. Each category will include a high level discussion covering the basics of what is needed, with applicable screenshots.

Hardware Selection and Builds

Before installing FreeNAS, it is important to make sure the hardware you are using will meet your needs. The following section will go over basic considerations when planning the hardware you'll use for your FreeNAS storage. A training course on FreeNAS Architecture and Performance Basics is available at <http://www.freenas.org/freenas-zfs-training/>.

Boot Devices: FreeNAS can be installed to at least one drive that is separate from your storage pool. FreeNAS will use the entirety of the drive, so it is generally recommended to utilize a flash device, such as a USB stick or SATA DOM, because standard hard drive capacities are well beyond what FreeNAS requires. Minimum size for your boot drive is 8GB, though 16GB is recommended so you can rollback to previous or alternative OS instances. You may also want to use two drives in mirrored configuration if you want additional data protection for your OS drive. Off-brand flash devices are generally not recommended due to their questionable reliability. Additional information can be found in the FreeNAS Documentation.

Memory: The general rule of thumb on memory is 4GB minimum and 1GB of memory per TB of storage. You will also want to consider whether ECC (Error Correcting Code) memory is important for your use case. ECC memory safeguards against lost write data in the instance of a system crash or power outage.

CPU: Because FreeNAS is based on FreeBSD, Intel processors are generally recommended, though AMD should also work just fine in most cases.

Cache Devices: Dedicated Cache Devices can help improve your system performance substantially. An L2ARC places your most frequently and recently used data in cache to improve your Read performance. Adding an SSD or other high performance flash device as your L2ARC is recommended.

Having a Separate Intent Log (SLOG) device for your ZFS Intent Log (ZIL) can help improve your write performance on synchronous write use cases, like database applications and NFS environments. As the ZIL flushes data every five seconds, performance is more important than capacity for a SLOG.

Check out our documentation on the Volume Manager for how to add L2ARC and SLOG/ZIL devices in FreeNAS: http://doc.freenas.org/9.3/freenas_storage.html#volume-manager.

HDDs: Desktop drives are not recommended for NAS storage. Enterprise or NAS rated drives, such as Western Digital Red drives, are calibrated to handle the load placed on them reliably.

You will also want to determine your desired level of disk redundancy for your storage pool. FreeNAS can be set up in striped, mirrored, both (RAID 10) or distributed parity configurations with up to three parity levels (RAID-Z1, RAID-Z2 & RAID-Z3). Mirrored pools will have half their raw capacity available as usable storage. In parity configurations, do keep in mind you will lose about a drive's worth of usable capacity for each parity level.

Network Interface: You will want to make sure your Network Interface Card meets your throughput needs and is compatible with your network speeds (you won't need a 10GbE interface card if your network switch is only 1GbE). You may also want to consider whether using a multiport NIC with link aggregation will help. You can read about types of aggregation in the FreeNAS documentation here >> http://doc.freenas.org/9.3/freenas_network.html#link-aggregations

In regard to NIC manufacturers, Intel cards are recommended for 1GbE connections and Chelsio cards are recommended for 10GbE.

Installation

You will first want to grab the latest version of FreeNAS from <http://www.freenas.org/download/>. Next you'll want to write the .iso file you've downloaded to disk or USB. You can read documentation about writing to flash devices in different operating systems at http://doc.freenas.org/9.3/freenas_install.html#preparing-the-media.

Once you've got the FreeNAS iso on your disk or flash device, you will want to insert it in your system and boot from the appropriate device to load the FreeNAS Installer.

Getting an IP address

You will want to have a keyboard and monitor attached to your FreeNAS hardware. As the device boots up, it will display text messages on the display as the operating system loads. When the system has finished booting, you will see a screen similar to Figure 1. *Note: Please be aware that all operations performed at this console menu are performed with root privileges and will override any existing settings.

```
Console setup
-----

1) Configure Network Interfaces
2) Configure Link Aggregation
3) Configure VLAN Interface
4) Configure Default Route
5) Configure Static Routes
6) Configure DNS
7) Reset Root Password
8) Reset to factory defaults
9) Shell
10) System Update (requires networking)
11) Create backup
12) Restore from a backup
13) Reboot
14) Shutdown

You may try the following URLs to access the web user interface:

http://192.168.1.119

Enter an option from 1-14: █
```

Figure 1: FreeNAS Console

If there is a DHCP server on your network, the Ethernet port on the device will automatically receive an IP address that can be used to access the device from a web browser. In the example shown in Figure 1, the device is reachable at <http://192.168.1.119>.

If your network does not have a DHCP server, refer to Section 3: Booting into FreeNAS in the FreeNAS® 9.3 Users Guide for instructions on how to manually set an IP address. This page also goes over the console menu options.

You may now unplug the USB keyboard and monitor from the back of the FreeNAS Mini, as they are not used for the rest of the configuration process. You are now ready to configure your FreeNAS® device.

Logging in for the first time

Direct your web browser to the IP address displayed at the console of your FreeNAS Mini. You will need to log in to the FreeNAS Web User Interface with a username and password. The username is always “root”. The default password is “abcd1234”.

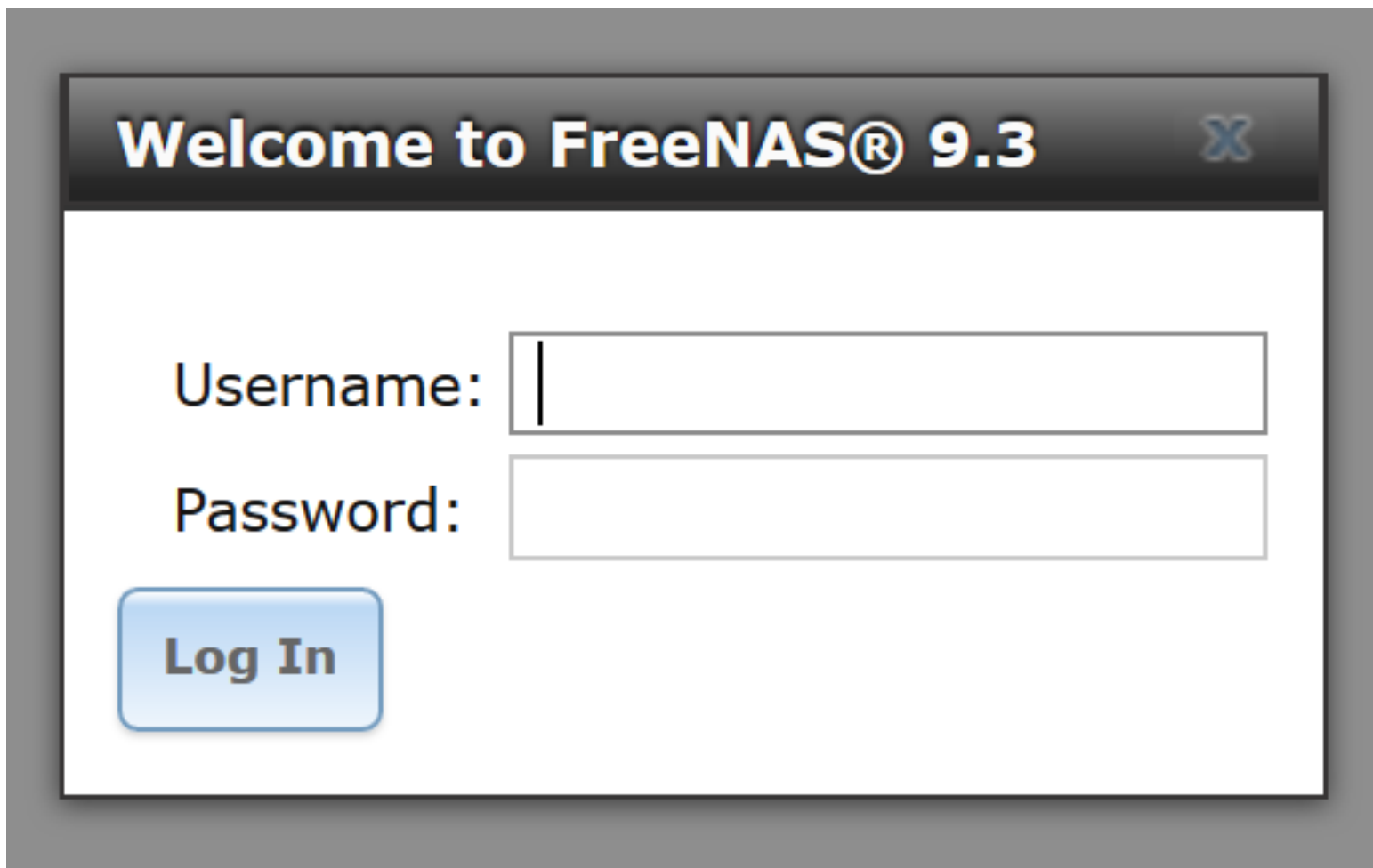
A screenshot of the FreeNAS 9.3 login window. The window has a dark grey title bar with the text "Welcome to FreeNAS® 9.3" and a close button (X) on the right. The main content area is white and contains two input fields. The first field is labeled "Username:" and has a single vertical line cursor inside. The second field is labeled "Password:" and is empty. Below the password field is a blue button with the text "Log In" in white.

Figure 2: Login Menu

Once you’ve logged in, the first thing you’ll want to do is change your ‘root’ password to something only you know. To do this, go to the User tab in the Account Menu, select the root user and then click on the “Modify User” button. This will open up a pop-up window to change your password. Just enter in your preferred password into the “Password” and “Password confirmation” fields, then click the “OK” button at the bottom of the menu.

Figure 3: User Menu

Modify User

User ID:

0

Username:

root

Primary Group:

wheel

Home Directory:

/root

Browse

Shell:

cs

Full Name:

root

E-mail:

Password:

Password confirmation:

Disable password login:

Lock user:

Figure 4: Modify User Menu

You have now successfully installed FreeNAS and secured your login information!

Conclusion

Hopefully, part 1 of this guide has provided you with all the information needed to get your FreeNAS system designed, built and installed. Next month's installment of the FreeNAS Getting Started Guide will review the Initial Configuration Wizard. Please check out the Additional Resources provided for more FreeNAS related guidance in the meantime.

Additional Resources

Blogs:

- FreeNAS Best Practices:

Part 1

<http://www.freenas.org/blog/a-complete-guide-to-freenas-hardware-design-part-i-purpose-and-best-practices/>

Part 2

<http://www.freenas.org/blog/a-complete-guide-to-freenas-hardware-design-part-ii-hardware-specifications/>

Part 3

<http://www.freenas.org/blog/a-complete-guide-to-freenas-hardware-design-part-iii-pools-performance-and-cache/>

Part 4

<http://www.freenas.org/blog/a-complete-guide-to-freenas-hardware-design-part-iv-network-notes-conclusion/>

- FreeNAS: A Worst Practices Guide <http://www.freenas.org/blog/freenas-worst-practices/>
- Forums: <https://forums.freenas.org/index.php>
- Videos:

Setting up your First NAS with FreeNAS https://www.youtube.com/watch?v=Otcke1QR9_U

FreeNAS 9.x Video Series <http://www.freenas.org/about/videos/>

- Documentation: <https://doc.freenas.org/>
- Webinar Training Courses:
Intro to FreeNAS course <http://www.onlinemeetingnow.com/register/?id=tit0ugmvdz>
- Webinar <http://www.freenas.org/freenas-zfs-training/>

USING FREEBSD AS A FILE SERVER WITH ZFS

In this course, we will learn how to use the current ZFS capabilities to help us build a home file server using FreeBSD 10.3.

Course launching date: 04th of July 2016

What will you learn?

- ZFS administration
- ZFS concepts and features

What skills will you gain?

- ZFS administration basics

What do you need?

- FreeBSD 10.3 with root privileges
- At least 10 GB free space

What should you know before they join?

- Basic FreeBSD administration knowledge



WORKSHOP

Module 1: FreeBSD and ZFS

Introduction to ZFS under FreeBSD

- Why ZFS on FreeBSD?
- ZFS features and concepts

Module 2 title: ZFS Administration

Module 2 description: Cover the commands and features to administrate ZFS volumes

- Create, destroy, list pools
- Zpools: single, mirrored, raid
- Understand ZFS properties

Module 3 title: Putting it all to work: Hosting our files using ZFS

Module 3 description: With the previous acquired knowledge, create a plan on how to organize our files and pools to host our files.

- Set ZFS properties based on the content of the files to host
- ZFS tuning
- Create a File Server using our pools

For more info visit our web page:

<https://bsdmag.org/course/using-freebsd-as-a-file-server-with-zfs-2/>

Don't hesitate to ask your questions at

marta.ziemianowicz@bsdmag.org



You do not need a big diploma to realize big projects, you need to be willing to go out of your comfort zone

Interview with Eric Turgeon, Founder and Leader of GhostBSD

by Marta Ziemianowicz, Marta Strzelec & Marta Sienicka



[BSD Magazine]: Hello Eric, how have you been doing? Thank you for agreeing to the interview. Can you tell our readers something about yourself?

[Eric Turgeon]: I am doing well, I am glad to answer your questions! I am a full time Inventory Controller Specialist for Kent Building Supply Distribution Center, a husband, and father of one son. I am also a founding member of the church Église Cité De La Grâce and the founder/leader of the GhostBSD project. I am a weight lifter and I do like downhill biking, my favorite sport is rugby. I am a BSD license and copy-free license advocate. I am a Canadian living in Dieppe City in NB and I speak French and English.

[BSD Mag]: What does it mean that you are a Project Leader for GhostBSD?

[ET]: Basically I am the founder of GhostBSD project, my leadership is not dictatorship, I try to be really open minded to all subjects and sometimes new features would be implemented without my knowledge. People contributing on GhostBSD have mostly the same power than I have over the project.

[BSD Mag]: There is absolutely no control over new features? How does that work exactly?

[ET]: I do trust the people that are officially in the GhostBSD team enough to let them work on new features; however, I still have the final say, but most of the features that

are added are not necessarily for the user. For example, they can be on the build tool that we use to build GhostBSD. I try to act like GhostBSD is Andrea's and Ovidiu's project as well, that is why they are free to work and improve on what they want.

[BSD Mag]: What is GhostBSD Project all about? And what is the difference between it and other BSDs? And why is it called Ghost?

[ET]: GhostBSD is born from the idea of having a Gnome FreeBSD, counterfeit of PCBSD which was KDE only when GhostBSD was born. GhostBSD comes from Gnome hosted by FreeBSD G host BSD, it is also why the G hostBSD logo on the website are distinct from the G host BSD, and the original pronunciation is G host BSD and not Ghost BSD, but even I call it Ghost BSD because it kind of sounds cool. GhostBSD is mainly a Desktop oriented OS, our focus is mainly to be a GTK alternative to PCBSD and being a bridge to the new FreeBSD user.

[BSD Mag]: What do you like the most about open source systems?

[ET]: I do like the fact that because FreeBSD is open source, and well documented, I did learn C programming, python, and UNIX shell programming, as well as learning how to build a whole FreeBSD system. My main opinion on open source is that it helped me to learn to program and I probably learned faster than with a closed source. I did play with the code of the open source program to see what would happen. With open code you cannot hide backdoors and spyware, someone will one day find out. Additionally, a lot people can work on the same piece of code around the world and make it work better and be more secure.

[BSD Mag]: So would you say that open source systems are generally more secure than closed source?

[ET]: In my experience yes, if I found a vulnerability I can find where it needs to be fixed and send a patch upstream, or point the developer to the faulty code. On Close Source, you can only send the vulnerability and wait for the company to fix it. On most of the big open source projects, the community responses fast and software gets fixed faster - most of the time.

[BSD Mag]: How is it to be a Leader of such project? What are your responsibilities and goals?

[ET]: Wow, I am not a great leader and never have been a leader, I am learning has the time passes. Unfortunately, with my explosive temper (when I am overwhelmed) I have lost some people on the project. I can say that Ovidiu was a great helper and probably the only one who has been on the project since almost from the start, and I owe him a lot. I did learn to make sure I understand people's point of view and ideas before replying, which have been helpful to avoid conflict. I do not want to be Linus Torvalds, therefore I try to do everything I can to understand others point of view. My leadership is only to provide a structure, a guideline, and keeping a balance on the project.

[BSD Mag]: If you had to give someone just one reason to use GhostBSD, what would it be?

[ET]: Oh, mmm, GhostBSD is FreeBSD made easy! I think that should be our slogan lol.

[BSD Mag]: How do you encourage people to contribute to GhostBSD?

[ET]: I do try to let people know that they can contribute to GhostBSD in any area they want to. GhostBSD is an Open Community and all our code, website, and wiki is open for everyone to work on.

[BSD Mag]: Is it hard to find a sponsor(s) for BSDs projects? Or are you mostly relying on money from donations?

[ET]: We are not going after sponsors, but we have a sponsor program that benefits us and our sponsors. All monthly donations and subscriptions become automatically a sponsor, also one time donations of over 100 CAD become sponsors too, this way we kind of give back to people who donate a fair amount of money. We started a Patronage program, but we did not put much effort in it yet, but there is more benefit to our user who becomes a patron to GhostBSD.

[BSD Mag]: What are those benefits?

[ET]: Depending on the level of the Patron donation, it can be: get a newsletter before publication, Get releases before the official release, sponsors badge, sponsors level (good for advertising), a GhostBSD sticker, and a shirt and more. And there are also goals to reach when GhostBSD gets to a certain amount per month, like move GhostBSD hosting on a VPS or a dedicated server, weekly news, full time development, financing people to go to Linux/BSD/Open Source conference, and much more. Patrons let us make it more manageable and accountable.

[BSD Mag]: Do you have any other favorite open source system?

[ET]: Mmm, not really. FreeBSD has been my favorite OS for over 9 years and GhostBSD has been born from FreeBSD and PCBSD. I did start to contribute a bit of time to DesktopBSD and most of GhostBSD Developers are contributing time to it, and we hope that people will love it. It is ironic because we were supposed to release a Gnome3 version of GhostBSD, and since DesktopBSD is releasing Gnome3, we might not release GhostBSD with Gnome3 version - we will see.

[BSD Mag]: GhostBSD 10.3 ALPHA1 is now ready for Testing. That is another way of contributing to the project. What are the other ways to help?

[ET]: Sure there is testing and reporting problems, but what I am looking for people to take care of is the documentation. I will be honest, our documentation is horrible, I am not a writer, and I

really have no time for writing the documentation. If the people who are reading this interview would like to contribute to GhostBSD, this would be the best place to start contributing. Our wiki is open to anyone to contribute, and if people need help, or information, the forums and IRC are the best place to contact us. There is also a code if people are in shell scripting, Python, C and etc. It is always appreciated to have more developers, also for people that cannot code, or do not feel comfortable writing documentation, there is a place for them too; they can engage with the community in IRC and the Forums, also a Bug Keeper will be needed soon.

[BSD Mag]: Can you tell us something about the license? How to get it and who can get it?

[ET]: GhostBSD is licensed under the BSD 2-clause license ("Simplified BSD License" or "FreeBSD License") which people can get at <http://www.ghostbsd.org/license>

[BSD Mag]: Do you have any tips for our readers?

[ET]: I did not graduate from high school, yes I was good in physics and algebra, but not in French grammar. I wanted to study computer science, but that dream was farfetched because of my academic grades in grammar. When I started to use FreeBSD, and started to learn how to change configuration in FreeBSD, I started to think Oh, this is almost like programming. I did start to learn programming by myself, I did start GhsotBSD at the same time. Today I do not consider myself a good programmer, but I am able to learn what I need to get the job done.

Most people say it is easy for you to learn programming if you use FreeBSD. No it is not easy, but when I started to use FreeBSD, I was 25 year old and I had only started to use computers two years earlier. Before that I did not know anything more than the basics of using the internet and installing software. I did not have any experience with a computer's OS's, kernel, etc. I started from the ground floor, with no knowledge in tech; however, I was curious and every step that I accomplished was empowering me.

You do not need a big diploma to realize big projects, you need to be willing to go out of your comfort zone, being willing to spend time to learn the tool you need to realize your projects, and focus on the important parts and if you fail, learn from it.

Barclays bank, as part of their Life Skills television and Internet campaign, are advising those entering the job market to use more professional email addresses. In light of their involvement in the Libor scandal, where they attempted to manipulate the benchmark inter-bank borrowing rate, can we take this advice seriously?

by Rob Somerville

There is an old joke about banks. A man asks for a loan, and checking his credit history, the bank manager says he will have to decline the application. However, as he is a gambling man, if the customer can tell him which one of his eyes is a glass prosthetic one, he will give him the loan. After thinking for a few seconds, the customer retorts "Your left eye", and the bank manager is shocked. Having won the bet successfully many times over the years, he agrees to giving the man the loan (at a higher rate of interest) provided he tells him how he worked this out. "Simple" the man says, "It was the eye with the most compassion in it".

Since working with Bela Hatvany and Mike Kidron in the 1980's at Global 1000, I have always been fascinated by the ethical, moral, PR and marketing stances organizations have taken. Global 1000, as part of their research, was examining the mergers and acquisitions markets, and also the green and ethical credentials of many Blue Chips. There has been a major move since then in the areas of repu-

tation management, and for companies to embrace a more ethical and morally engaged approach to business. Nowhere is this more prevalent than in the banking sector, which has been hammered with scandal after scandal and the PR mood music is very much "We are nice guys, really, and have learned our lesson, how can we help you"?

This whole campaign truly sticks in my throat. Apart from the blatant commercial opportunism that banks traditionally exert in trying to capture young customers for life, especially in the later years of secondary schools, and then at colleges and universities, I don't think banks in general understand technology they are playing with other than being a tool to bring efficiency, cut costs, and maximize profit. Or maybe they do. A significant meme on the Internet is the concept that we are heading towards a cashless society, potentially with each one of us having a chip embedded or a some form of bar-code tattooed upon our being.

While this vision might be the domain of right wing crazies and religious fundamentalists, from a commercial standpoint, a cashless society would certainly reduce fraud, but in what proportion to electronic fraud, tax shelters and good old fashioned hacking would be an interesting debate indeed.

The culture of the Internet and the banks seems to be at odds with each other. A while back, a particularly well crafted phishing email ended up in my in-box. The website was still live, so if I were uneducated in such matters I could easily have mistaken it for a genuine email if I was the customer of that particular bank. Concerned, and as I would not like to see anyone ripped off (even the banks if that were actually possible), I phoned their head office to report it. Being a weekend, I was informed there was no security team active over the weekend that would be able to address my enquiry. The phishing site was still live over the weekend until a take-down early Monday morning. How much customers lost financially or in way of confidential data is unknown, but the lack of an adequate channel to report such an incident deeply concerns me.

With the recent SWIFT attacks that have defrauded the industry of millions, the industry is slowly getting its head round Internet fraud. So much so, under proposed changes proposed by banks and the UK government, customers may have to foot the bill for fraud against their accounts. This level of fraud, which is approaching almost three quarters of billion pounds per year, is not chump change. Maybe I am just a cynical old goat, but it is mighty convenient that an industry that has

not moved forward in adequate long term IT investment over the years (other than short term cost cutting to burnish the bottom line for shareholders) wants to kick the problem downhill to the customer now that the e.coli infected brown sticky stuff has hit the ventilation device.

What really yanks my chain here though is how a business sector having a reputation for lack of accountability and the credibility approaching that of a common criminal has the sheer gall and arrogance to lecture youngsters on what their email address should be. They forget that on the Internet – like in the days of CB radio – everybody has an alter identity, a handle if you prefer. Sure, if you want to go and join a professional organization it is better to have a professional email address. The message itself is not particularly offensive, it is common sense really. I would not want to communicate with an accountant at scammeseless@ripoff.ng for instance. However, there are industry sectors that are much more open to creativity and are crying out for young talent where this is less of a factor. And if we are truly objective, the cultural move is more towards social media than corporate email. Certainly my teenage daughter laughs at email as being old hat. And I am really uncomfortable with the concept of a bank being an “educator”, especially as Matt Barrett, the chief executive of Barclays until 2006, admitted he did not use his a credit card to borrow money because it was too expensive – a message he had been urging his four children to adopt. Now that would have made a good PR campaign for teenagers.

By engaging with such personal dialogue, this PR campaign is close to the territory of “get your hair cut, remove the nose piercing, cover up that tattoo” otherwise you will not be accepted and will not get a job. Which is all very nice and good, a classic middle class mantra, provided there were jobs and opportunities out there. So we have a very corporate, sensible message relayed by a sector that causes more pain and suffering to thousands, if not millions of individuals by the fact that they are too big to fail and are a global player with all their fingers in the pie. Depending on where you rest on the political and moral spectrum, this campaign lands between propaganda, through sheer hypocrisy to wickedness.

I wouldn't be offended so much by this futile attempt at social engineering if I had not had the pleasure to work at an international bank for a time and also attend interviews in the city of London for roles in the sector. The whole culture is based on judging a book by its cover, whether or not you wear the right tie, or don't have a beard. I fail on both counts. The arrogance in the industry is endemic. Having worked with the phenomenal team at Global 1000 (whose humility I will take to my grave), I believe I have had a lucky escape. My email address is linuxgreybeard@gmail.com. I have a grey beard and love Linux (and *BSD of course). If my email address offends your sweet, tender sensibilities, I'd really rather not hear from

you.